國立臺灣大學電機資訊學院資訊工程學系

碩士論文

Department of Computer Science and Information Engineering
College of Electrical Engineering and Computer Science
National Taiwan University

Master's Thesis

一種區塊鏈上無收據的投票協議

A Receipt-free Protocol for Voting on Blockchains

陳冠廷

Timothy Chen

指導教授：洪一平 博士

Advisor: Yi-Ping Hung, Ph.D.

中華民國 113 年 07 月

July 2024

doi:10.6342/NTU202404046

國立臺灣大學碩士學位論文
口試委員會審定書
MASTER'S THESIS ACCEPTANCE CERTIFICATE
NATIONAL TAIWAN UNIVERSITY

一種區塊鏈上無收據的投票協議

A Receipt-free Protocol for Voting on Blockchains

本論文係陳冠廷君（學號 R10922188）在國立臺灣大學資訊工程學系完成之碩士學位論文，於民國 113 年 7 月 24 日承下列考試委員審查通過及口試及格，特此證明。

The undersigned, appointed by the Department of Computer Science and Information Engineering on 24 July 2024 have examined a Master's thesis entitled above presented by TIMOTHY CHEN (student ID: R10922188) candidate and hereby certify that it is worthy of acceptance.

口試委員 Oral examination committee:

洪一平	黃秀男	楊佳玲
（指導教授 Advisor）		
彭世偉		

系主任/所長 Director: 陳祝嵩

中文摘要

投票是現代民主的重要過程，然而自然會面臨強迫和腐敗的問題。隨著區塊鏈技術的興起，在區塊鏈上進行此類活動因其能提供安全、透明且去中心化的投票系統而引起關注。然而，由於區塊鏈本質上是一個公開帳本，這樣的設計將導致關於投票人隱私的問題隨之而來。

本文中，我們提出了一個使用區塊鏈作爲公開計票的無收據投票協議。通過將操作權限分離爲負責驗證選民身份的身份機構和負責生成加密選票和計票的投票機構，我們確保在保護選民隱私的同時，允許任何第三方驗證投票和計票的完整性。該協議利用同態加密進行計票狀態，並使用RSA簽名進行選票驗證。在假設兩個主辦機構都是誠實的情況下，該協議實現了無收據的特性。

關鍵字：區塊鏈; 電子投票; 隱私

doi:10.6342/NTU202404046

Abstract

Voting, an important process of modern democracy, is naturally subject to coercion and corruption. With the rise of blockchain technology, conducting such campaign on blockchain has drawn attention with its potential to provide a secure, transparent and decentralized voting system. However, with the nature of blockchain being a public ledger, concerns about the privacy arise.

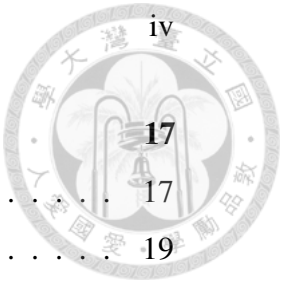
In this work, we present a receipt free voting protocol utilizing blockchain as public tally. By separating operating authority into identity authority which is responsible for verifying the voter's identity and voting authority which is responsible for generating encrypted ballots and tallying, we ensure that the privacy of voters is protected while allowing any third party to verify the integrity of the vote and the tally. The protocol make use of homomorphic encryption for the tally state and blind signature for ballot verification. The protocol achieve receipt-freeness assuming that both authority is honest.

Keywords: Blockchain, Electronic-Voting, Privacy

doi:10.6342/NTU202404046

Contents

doi:10.6342/NTU202404046



doi:10.6342/NTU202404046

List of Figures

doi:10.6342/NTU202404046

Chapter 1

Introduction

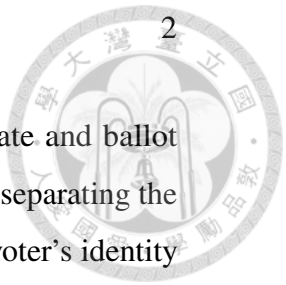
Voting, being a critical process of decision-making, naturally subject to coercion and corruption. With the rise of blockchain technology, conducting such campaign on blockchain has drawn attention with its potential to provide a secure, transparent and decentralized voting system. However, with the nature of blockchain being a public ledger, concerns about the privacy arise.

The privacy of voters has been a major point of research for electronic voting. While unlike traditional voting with paper ballots, electronic voting provides benefits such as convenience and accessibility, it also brings a challenge of keeping votes secret. A trivial solution is to mimic the traditional voting system, which "shuffle" the votes. By mixing a batch of inputs before output, the correspondence between the input and the output is weakened. Another solution is make use of homomorphic encryption, allowing votes to be cast and tallied without revealing the content of the vote.

A concept, called receipt-free, is introduced as a result of the challenge[1]. It meant to both protect the voter's privacy and prevent vote-buying and/or coercion. A receipt free protocol means that the voter, whether willing or not, cannot proof to others how his/her vote is cast.

In this work, we propose a receipt-free protocol for voting on blockchains. While benefiting from the natural transparency and decentralization of using blockchain

doi:10.6342/NTU202404046



as a tally, we combined homomorphic encryption for the tally state and ballot generation, and blind signatures for voter's identity verification. By separating the operating authority into two parts based on their functionality, the voter's identity cannot be disclosed by any single authority, and the system is receipt-free assuming that both authorities are honest. In addition, any interested third party can verify the integrity of the vote and the tally, and unlike most existing work, the protocol is designed so that the voter only needs to participate in the voting process once.

doi:10.6342/NTU202404046

Chapter 2

Related Work

2.1 Receipt-Free Electronic Voting

A receipt, in the context, is effectively a transferable proof of how a voter voted. Receipt-free means a protocol, under certain assumptions, generate no such receipt/evidence, which thus prevent vote-buying and/or coercion.

This concept was first introduced in 1994, while Benaloh and Tuinstra [1] utilizing their own homomorphic encryption [2] and an interactive proof based on the system under the assumptions of a secure ”voting booth”, which provide one-way not-recordable secret communication. Their work include two kinds of protocols, one with a single trusted authority and the other with multiple authorities to distribute the risk. The single authority version is receipt-free if the authority is honest while the multiple authority version, later pointed out by Hirt et al. [3] had a flaw that allowed voter to generate a receipt. Specifically, if the voter make a commitment, which he/she is able to, of how each encrypted ballot is ordered prior to proving his vote is valid, this commitment will work as proof of the direction of the ballot with as it work in the single authority version. Similar works like [4, 5] also make use of homomorphic encryption to achieve secrecy of ballots.

Early works like [6] instead of homomorphic encryption implement shuffling mechanism to disguise the voting direction of voters. By rearranging the order of

doi:10.6342/NTU202404046

multiple inputs, the linkage between the voter and the votes is weakened. Those work also make use of different interactive proof system to ensure the correctness and verifiablity of the vote while keeping the privacy of the voter. Following works such as [7, 8, 9, 10]also implement similar mechanism.

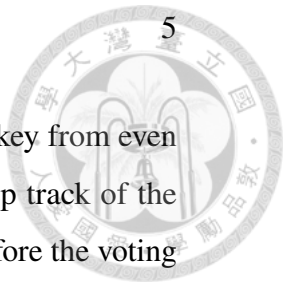
2.2 Voting on Blockchain

Blockchain, being a trusted and decentralized computation and storage platform, has the potential to serve the critical part, both as a computing device and a storage, of electronic voting system. Blockchain's advantage in transparency and data integrity has seen its application in field such as credential and certification[11, 12]. With the rise of blockchain technology, voting on blockchain has drawn attention with its potential to provide a secure, transparent and decentralized voting system [13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23]. Specifically, a mechanism called smart contract, which is a public program that is public and can be called on the blockchain has the potential to serve as a trusted third party during the voting process. However, an obvious challenge is the privacy of the voters, as being a public ledger, each and every transaction and state on the blockchain is public.

One approach is changing the form of vote itself. As proposed by [22], by voting as a distribution, a user can easily distribute his/her vote without revealing the direction of the vote. Their protocol then circulate those distributed votes and then sum them up to get the final result. Although it achieved privacy and verifiablity without a trusted third party, the protocol requires users to actively participate in multiple stages of the voting process even in ideal conditions (Everyone is honest). Additionally, distribution voting multiplies the number of votes. These make the protocol to be suitable for small scale occasions, such as DAO mentioned in the paper, but not for large scale campaigns.

Another approach is to assume a secret holding device that can keep the key or secret to be used in only specific ways in specific time. Dimitriou's approach

doi:10.6342/NTU202404046



[15] assume a device $\mathcal{TR}$ that keeps the user's choice and signing key from even the user's access. They use a zkSNARK system from [24] to keep track of the eligibility of each voter. In addition, to keep the tally state secret before the voting period ends, votes are encrypted by the voter with $\mathcal{TR}$, and voters are required to submit the encryption key to the tally after the campaign. Receipt-freeness is achieved by keeping the receipt within the special device $\mathcal{TR}$ and destroyed after the campaign, however, it still requires the voters to actively participate in two stages of the voting process.

Work from [25] takes an approach utilizing homomorphic encryption to keep the privacy of the vote. Their protocol is designed that each voter encrypt their choice of vote and submit to a group of volunteer servers, and those servers send those batch of votes with zero knowledge proof of correctness and eligibility to the blockchain. Their protocol distributes the private key of the homomorphic encryption to another group of volunteer, which are to decrypt the tally. This prevents a single entity to decrypt the tally and reveal the vote by distributing trust. However, though the protocol is claimed to be receipt-free, as it ask for voters to encrypt their vote, and a vote is verified on the public blockchain, a receipt can be formed as a combination of the generating process of the encrypted ballot with proof, and the verification process on the blockchain of the specific encrypted ballot. This flaw is somewhat similar to the one pointed out by Hirt of Benaloh's multiple authority protocol.

doi:10.6342/NTU202404046

Chapter 3

Protocol

In this chapter, we present the proposed scheme for a receipt-free electronic voting system in a general setting, without specifying the underlying cryptographic primitives. We first give assumptions and requirements of the system, and then describe the proposed protocol. Last, we provide an open-source proof-of-concept implementation of the protocol.

3.1 Tool used

3.1.1 Signatures

In the protocol, RSA signature is used to sign both the ballots and the selectable choices, and both authority should construct RSA key pair and public their public key prior to the campaign. The signature for the ballot is to show the eligibility to vote without revealing voters' identity, and the signature for the choices is to show the choices are generated by the voting authority.

3.1.2 Homomorphic Encryption

The system uses an encryption scheme, so the votes can be submitted and counted on public blockchain without revealing the content of the votes. This is to hide the

doi:10.6342/NTU202404046

partial result on the blockchain during the campaign, as well as to present linkability between the vote's direction and the voter. For the purpose, these properties of the encryption scheme is required:

- A instance of the encryption scheme consists of a encryption function E, a decryption function D and a certification function D' with parameter N, r.
- An encryption z of x can be formed with E with $0 \leq x < r$.
- There are two functions $\otimes, \oslash$ such that $z_1 \otimes z_2 \in E(x_1 + x_2)$ and $z_1 \oslash z_2 \in E(x_1 - x_2)$.
- A unique x and a certificate u can be generated from z with D and D' respectively, with $0 \leq x < r$ and u can be used to show that z is a valid encryption of x.

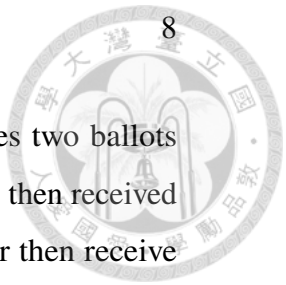
3.1.3 Public Blockchain

In this protocol, the blockchain is used as a public ledger to store the encrypted votes and results. We assume no linkage between the address used to submit the votes and the voter's identity. This can be achieved by using a new address for each vote, or using a mixing service to hide the linkage. In addition, we assume all transactions and state on such blockchain are public and is reliable with any consensus mechanism used.

3.2 System Overview

The proposed method is mainly a combination and improvement from TAVS[26], addressing their privacy from their original Soldity implementation[27], and the flow of the system is shown in Figure 3.1. The protocol consists of two stage, and requires voters to interact with two separate authority instances a public blockchain tally with the idea of separation of identity verification and ballot decryption from

doi:10.6342/NTU202404046



[26]. The first stage is the voting stage, where the voter generates two ballots with different directions and register with the identity authority(IA), then received the required information. From the voting authority(VA) the voter then receive commitment to distinguish the direction of two votes. After that the voter sends the selected ballot to the blockchain tally with an anonymous account (address), and the tally will calculate the encrypted result. The second stage is the tallying stage, which only involves the voting authority and the blockchain tally. The tally will decrypt the result and public the result and proof to the voter. This make voting for voters a one-step process, as they only involves in the first stage, different from most of the previous works.

doi:10.6342/NTU202404046

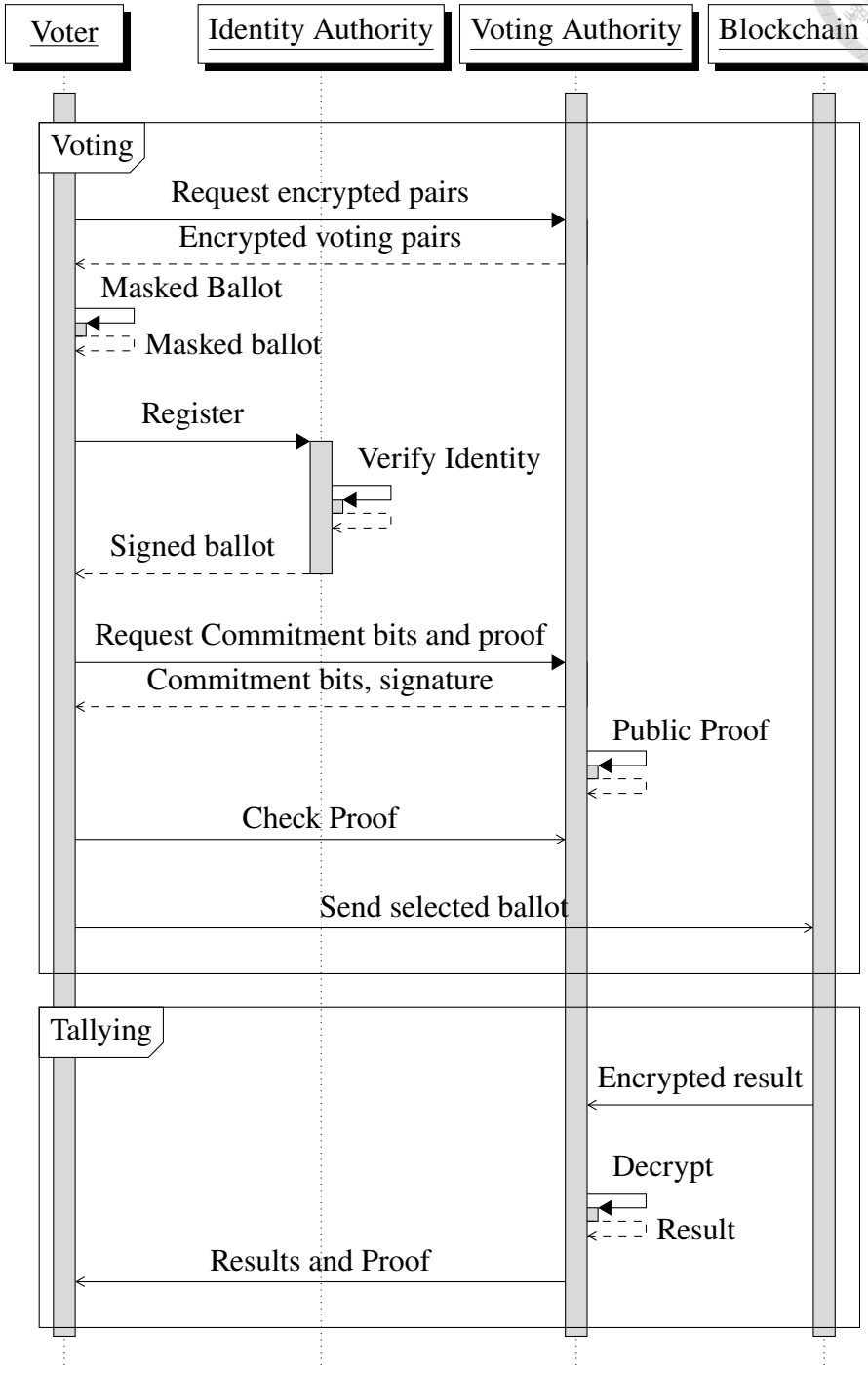


Figure 3.1: System Diagram

doi:10.6342/NTU202404046

3.3 Voting stage

3.3.1 Ballot Generation

The process of generating a ballot consists of two parts - the genration of unsigned ballot and the verification of those ballots. This process also verified the identity of the voter, however the direction of votes are not yet decided, and thus cannot be linked to the voter. In the beginning of voting process the identity authority public its RSA key pair with the voting authority providing homomorphic encrypted voting pair sequences generated by Algorithm 1, the implementation takes use of Benaloh's encryption scheme [2] with the correction from [28]. Here r, n, y is then released to the public, while p, q is kept as secret key. The sequence is generated so that each pair is formed with an element from $E(0)$ and an element from $E(1)$ in random orders.

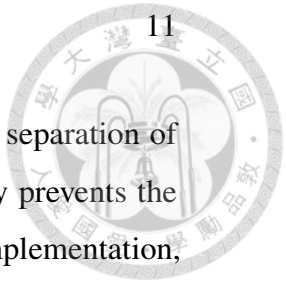
Then, a mask and its inverse is generated on voter's side privately using Algorithm 2. The two resulting ballots are of the two direction for the campaign. Notice that til now, the voter's cannot tell the direction of the votes.

After that, the voter interacts with the authority to verify the ballots. The identity authority will verify the identity of the voter and sign both ballots with its key with Algorithm 3. In this part, the authority will sign both way of the masked ballots using its RSA private key.

After receiving the signed ballot, the voter provides those ballot to the voting authority as a proof of eligibility for voting. The voting authority privately transmit the commitment bits and signature of both ballot to the voters so that the voter can distinguish the two ballots after the voters received the signed ballot and vote with them. A proof is then public by the voting authority to show that the commitment bits and the encrypted pairs are formed correctly. Both commitment and proof is generated with Algorithm 4.

The design here has two important roles. First is to maintain the so-called two-folded property. In their Soldity implementation [27], there is a concern

doi:10.6342/NTU202404046



in the privacy protection of the voters. Their design requires the separation of the verification of identities and counting of votes. This property prevents the authority from linking the votes to the voter. However, in their implementation, the unmasking of ballot is done on blockchain which is a public ledger. This significantly degrade the property of their original design.

In our design, the counting of votes is done under homomorphic encryption, and both side of the ballots are signed by the authority. Also, a signature from the voting authority is required to verify that the ballot contains a valid choice generated by the voting authority.

The voting authority also public an interactive proof for the whole sequence with the beacon output, just as Benaloh described in their original protocol.

3.3.2 Voting

The voting process is simply done by submitting the signed ballot to the blockchain tally. The tally then decrypt the result and calculate the result under homomorphic encryption with Algorithm 5. To ensure that the ballot is valid, the tally first check if the ballot is duplicated, then check if the ballot is signed by identity authority, then finally check if the hash against the mask and the underlying ballot is correct. In addition, it will check the choice selected by voter is generated by the voting authority by verifying the signature of the voting authority. This is to ensure that voter does not use a value generated by a third party since the encryption method is public. A choice generated by a third party will both reveal how the voter voted and disturb the result if it is not either an element in $E(0)$ or $E(1)$.

If the ballot is valid, the tally will update the state $s' = s \otimes v$ which in the Benaloh encryption scheme is multiplication.

doi:10.6342/NTU202404046

Algorithm 1: Voting sequence generation

Input: Homomorphic encryption function arguments r, N

Output: A sequence of encrypted pairs $\{(\alpha_0, \beta_0) \cdots (\alpha_n, \beta_n)\}$

begin

  $r \leftarrow$ A prime close to r

  $p \leftarrow$ A prime of size N such that $r \mid (p-1) \wedge r^2 \nmid (p-1)$

  $q \leftarrow$ A prime of size N such that $r \nmid (q-1)$

  $n \leftarrow p \cdot q$

  $y \leftarrow$ A random number in $\{1, 2, \cdots n-1\}$ such that $y^{(p-1)(q-1)/r} \mod n \neq 1$

  $E(m) \leftarrow \{x^r y^m \mod n | x$ is a relative prime in $\{1, 2, \cdots n-1\}\}$

  for $i \leftarrow 0$ **to** n **do**

    $a_i \leftarrow$ A random element in $E(0)$

    $b_i \leftarrow$ A random element in $E(1)$

    $\alpha_i \leftarrow \max(a_i, b_i)$

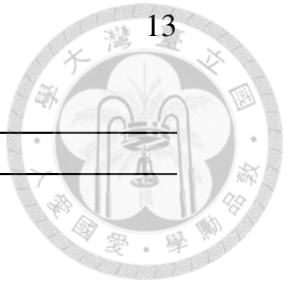
    $\beta_i \leftarrow \min(a_i, b_i)$

  end

  return $\{(\alpha_0, \beta_0) \cdots (\alpha_n, \beta_n)\}$

end

doi:10.6342/NTU202404046



Algorithm 2: Voter-Side Ballot Generation

Input: A hash function H

Input: IA's RSA public key $pk = \langle n_{RSA}, v_{RSA} \rangle$

Output: *Two* masked ballots $v_{pre}, \tilde{v}_{pre}$, mask and its inverse m, m^{-1}

begin

  Fetch an encrypted pair sequence $\{(\alpha_0, \beta_0) \cdots (\alpha_n, \beta_n)\}$

  $m \leftarrow$ A random number such that $gcd(m, n_{RSA}) = 1$

  $m^{-1} \leftarrow$ The modular inverse of m

  $h \leftarrow H(\alpha_0 || m)$

  $\tilde{h} \leftarrow H(\beta_0 || m)$

  $v_{pre} \leftarrow (\alpha_0 || h) \cdot (m^{v_{RSA}} \mod n_{RSA}) \mod n_{RSA}$

  $\tilde{v}_{pre} \leftarrow (\beta_0 || \tilde{h}) \cdot (m^{v_{RSA}} \mod n_{RSA}) \mod n_{RSA}$

  return $(v_{pre}, \tilde{v}_{pre}, m, m^{-1})$

end

Algorithm 3: Ballot Verification

Input: authority's RSA key $pk = \langle n_{RSA}, v_{RSA}, s_{RSA} \rangle$

Input: The voter's identity

Input: The two masked ballots $v_{pre}, \tilde{v}_{pre}$

Output: The signed ballots $v, \tilde{v}$

begin

  if *Vaild non-repeated identity* **then**

    $v \leftarrow (v_{pre})^{s_{RSA}} \mod n_{RSA}$

    $\tilde{v} \leftarrow (\tilde{v}_{pre})^{s_{RSA}} \mod n_{RSA}$

    return $(v, \tilde{v})$

  else

    return *null* `// Invalid identity`

  end

end

doi:10.6342/NTU202404046

Algorithm 4: Commitment bits and proof generation

Input: Beacon output $\{b_1 \dots b_n\}$ (Used for interactive proof)

Input: Prime p, q and r, n, y from Algorithm 1

Input: Encrypted pairs $\{(\alpha_0, \beta_0) \cdots (\alpha_n, \beta_n)\}$

Output: Proof and Verification pairs $\{\mathcal{D}_1 \cdots \mathcal{D}_n\}$

Output: Commitment bits $c_0 \dots c_n$

Function `Decrypt(`z`)`
- $x \leftarrow (zn^{-1})^{(p-1)(q-1)/r} \mod n$ *is equal to* 1
- $u \leftarrow (zy^{-x})^{r^{-1} \mod ((p-1)(q-1)/r)} \mod n$
- $s \leftarrow$ a uniformly chosen relative prime to n in $\{1, 2, \cdots n-1\}$
- $u \leftarrow us^{(p-1)(q-1)/r} \mod n$
- **return** (x, u)

begin
- **for** $i \leftarrow 0$ **to** n **do**
  - $c_i \leftarrow (\alpha_i n^{-1})^{(p-1)(q-1)/r} \mod n$ *is equal to* 1
- **end**
- **for** $i \leftarrow 0$ **to** n **do**
  - **if** b_i **then**
    - $\mathcal{D}_i \leftarrow$ (`Decrypt(`α_i`)`,`Decrypt(`β_i`)`)
  - **else**
    - **if** α_0 *and* α_i *decrypt to same value* **then**
      - $\mathcal{D}_i \leftarrow$ (`Decrypt(`$\alpha_i\alpha_0^{-1} \mod n$`)`,`Decrypt(`$\beta_i\beta_0^{-1} \mod n$`)`)
    - **else**
      - $\mathcal{D}_i \leftarrow$ (`Decrypt(`$\alpha_i\beta_0^{-1} \mod n$`)`,`Decrypt(`$\beta_i\alpha_0^{-1} \mod n$`)`)
    - **end**
  - **end**
- **end**

end

doi:10.6342/NTU202404046

Algorithm 5: Blockchain Tally - Receiving Vote

Input: Public key of IA's RSA $pk_i = \langle n_{RSA}, v_{RSA} \rangle$ and VA's RSA pk_v

Input: Public part of VA's Benaloh encryption system r, n

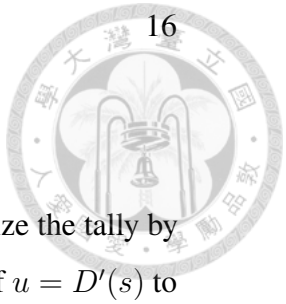
Input: Signed ballot v, mask and inverse m, m^{-1}, hash function H and VA's signature sig

Input: previous state s

Output: New state s'

```
begin
    if v is not duplicated and sig is valid signature of v by pk_v then
        v ← (v m^{-1})^{v_RSA} mod n_RSA
        v', h ← split v
        if H(v'||m) ≠ h then
            return s // Invalid ballot
        end
        s' ← s v' mod n
    else
        return s // Duplicated ballot
    end
end
```

doi:10.6342/NTU202404046



3.4 Tallying Stage

At the end of the voting time window, the voting authority can finalize the tally by sending the decryption $d = D(s)$ of the current state s with the proof $u = D'(s)$ to the blockchain tally. The tally can check if the decryption is correct by checking if $u^r y^d = s \mod n$. The public can then see the verified result on the blockchain.

3.5 Implementation

All algorithms' implementation in this chapter can be found on repository `github.com/timothychen1999/blockchain-vote-imp`. Both authorities' functionalities are implemented using Python 3. The blockchain tally is implemented on Tezos with LigoLang, a language for smart contract on Tezos. For testing purpose, a deployment contract is also originated on Tezos Ghostnet and is at address *KT19uWxxKusKywu45aYi9DBahssHh1JPerew*. The contract can be called to delpoy a new tally with admin set to the caller. The contract also provides entrypoints to receive votes and finalize the tally. A simplified version of sample vote transactions can also be found going through contract at address *KT1Db9AHAHAVSEHrCzZDDby9H4FBWwAsZ99M*.

doi:10.6342/NTU202404046

Chapter 4

Analysis

In this chapter, we first analyze properties of the proposed protocol. Then we discuss the possible threat that the protocol may face.

4.1 Correctness and Verifiability

The correctness of the protocol can be separate into two part, the voting process such that each eligible voter can cast one and only vote, and the tallying process such that each and every valid ballot is counted correctly. For the first part, during the ballot generation process, an eligible voter can only get two ballots signed by the identity authority, which are the only two that will be accepted by the tally. Although the voter may choose to cast both of the ballots, since the ballots are designed to be in two opposite directions, that can just be considered as an invalid vote.

For the second part, the correctness can be guaranteed by jointly considering the correctness of the two method. First, since we make use of the TAVS's ballot signing method, with the property of RSA signature and the marginal possibility of hash collision, we can assure that the choice of the voter is not altered during the transmission. For integrity, we briefly showed the process of the step. Let c be the voter's choice, m, m^{-1} be the mask and its inverse, n, v, s be the RSA component

doi:10.6342/NTU202404046

and H being the hash function, we can see that

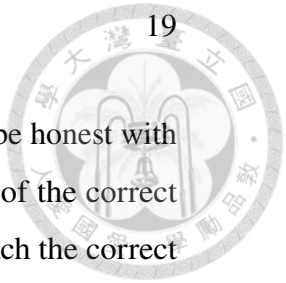
$$\begin{aligned} \text{Ballot} &= ((c||H(c||m)) \cdot m^v)^s && \mod n \\ &= (c||H(c||m))^s \cdot m && \mod n \\ \text{Decoded} &= ((c||H(c||m))^s \cdot m \cdot m^{-1})^v && \mod n \\ &= (c||H(c||m)) && \mod n \end{aligned}$$

with $H(c||m)$ to ensure the correctness of the given mask and choice. Additionally, since the removal of the mask is done on the blockchain, a voter can easily verify and proof that the ballot is correctly formed by performing the process on their own before submitting it to the tally, this ensures that a forge attempt from the identity authority is easily detected.

Then is the commitment and proof system that represent the direction of each ballot, which is similar to the single authority voting scheme proposed by Benaloh. For a third party's perspective, it can verify that the encrypted pair is formed correctly with the public proof as follows. Recall that when requesting encrypted pairs, the voter receive N extra encrypted pairs in addition to the one that is use as ballot ($(\alpha_0.\beta_0)$). For each extra pair (α_i, β_i), the voting authority, regarding random bits output by the beacon, either open it by decrypting a pair with proof showing that it consist exactly a 0 and a 1, or connect it by showing that $\alpha_i = \beta_0 \wedge \alpha_0 = \beta_i$ or $\alpha_0 = \alpha_0 \wedge \beta_0 = \beta_i$ with the operator $\oslash$. Therefore, if the voting authority construct a faulty pair for $(\alpha_0.\beta_0)$, it must do the same to every and only connected pairs. From a voter's perspective, the direction of both ballot indicate by the commitment bits can be verified since if the voting authority willing to construct an inverted commitment, it must also invert the bit on all connected pairs too. Both action requires the voting authority to predict N random bits output by the beacon correctly, giving a success rate of 2^{-N} which is unfeasible.

Next is the tallying process conducted on the blockchain, which is a simple summation under the homomorphic property of the Benaloh cryptosystem. Since we already show that the vote's choice do represent the voter's intention, and the

doi:10.6342/NTU202404046



choice is not altered during submission, and the blockchain should be honest with its native properties, the final state on the ledger is an encryption of the correct result. Even if the blockchain's consensus mechanism failed to reach the correct result, a third party can easily verify and potential correct the result by public doing each step of the tallying process. Finally, the decryption of the tally is proved and verified by the blockchain after the voting authority submit the decrypted result with its proof, which can also be repeated by a third party.

4.2 Privacy and Receipt-Freeness

The protocol is receipt free assuming that both authority is honest (not corruptible in the model) and the voter is only able to reach both authority during the ballot-generation process. The receipt-freeness can be shown by providing the "Coercion Strategy" [9] as inverting the receipt (the commitment bits) to the direction the coercer wants. Here we show that this protocol is receipt-free using the definition and symbols proposed by Moran et al[9].

Specifically, for the real world adversary $\mathcal{A}$ that ask a voter voting v to votes against his/her will, the voter can vote as is, and respond for $\mathcal{A}$'s query for proof with an inverted commitment crafted in the same way mentioned in last section, with the only difference is that he/she is crafting this afterward, so no prediction is needed. Since the vote send to the tally is not changed, it yields indistinguishable result as the ideal world $\mathcal{I}$. However, a coercer can still force an abstention by coercing the voter to cast both ballots.

In addition, the protocol is designed so that none of the authority can reveal a voter's choice on it own. The direction of each ballot is hidden under the private key of Benaloh cryptosystem, which is held by voting authority. The linkage between the ballot and the voter is cut off by the identity authority, which signs both direction of the ballot.

doi:10.6342/NTU202404046

4.3 Robustness

The robustness of the protocol is the ability to with partial corruption or sabotage without halting the whole system. This can be divided into two parts. First, in case of corruption of either authority, though the protocol is capable of detecting such corruption and halt the voting process given that both RSA signature and encrypted ballot is verifiable by both the voter and the third party, the campaign is and should be stopped. A corruption of the blockchain tally is unlikely to happen, however, if it does, any third party can easily replace that position and calculate the correct result because each and every transaction is public.

As for sabotage, only the identity authority is vulnerable to such attach. For the tally, since sabotaging a blockchain system would require to sabotage all of its active node, which is unfeasible, attack against that part of the system is unlikely to happen. For voting authority, multiple instances can be easily deployed to prevent such attack, as this part of the system is stateless, and the only thing that need to be share between is the private key of the Benaloh cryptosystem.

4.4 Other Threat

Still there are some vulnerabilities that the protocol may face. First, just as many other discussions on blockchain voting, voter identity verification is out of the scope. In this case, this relies on the identity authority. This means a coercer can force the identity authority to forge false identity, which can be used to cast multiple votes.

In addition, a coercer is still able to reveal the state of the tally before the voting period ends by coercing the voting authority, and can than reveal the direction of each ballot. However, the voter's identity is still kept secret, as the linkage between the ballot and the voter is cut off by the identity authority.

Last, like most if not all remote voting system, the protocol cannot provide any protection if the coercer can physically access the voter's terminal device during

doi:10.6342/NTU202404046

voting.

doi:10.6342/NTU202404046

Chapter 5

Conclusion and Future Work

In this work, we proposed a receipt-free protocol for voting on blockchains. By combining the homomorphic encryption for tally state and separating identity verification to a different entity signing both direction of the ballot, we ensure that correspondence between the voter and his/her vote's direction is cut off. In addition, the protocol is designed so that the voter only need to participate in the voting process once, and any interested third party can verify the integrity of the vote and the tally. As describe in analysis, the protocol itself is receipt-free and Verifiable, and is robust against partial corruption or sabotage.

There are still possible improvement though. First, as identity verification is still a non-decentralizeable single point of failure, a more robust identity verification system can be established. This can be done by utilizing a decentralized identity system, such as the one proposed by W3C, which allows a user to have a self-sovereign identity. In this way, other technique to allow anonymous identity verification, such as zk-SNARK or semaphore can be used as a replacement of IA's functionalities to construct a more secure and robust system.

Also, the protocol is vulnerable with a dishonest voting authority that reveal the private key of the homomorphic encryption. This risk can be mitigated by utilizing a threshold encryption system, which would require multiple malicious entities to collude to reveal the private key.

doi:10.6342/NTU202404046

Last, security measurement for terminal device used for voting could be taken into consideration. Possible solution such as asking voters to vote at a designated location with a secure device, or implementation based on security-purposed chips/hardware can be applied based on the requirement of the voting campaign.

doi:10.6342/NTU202404046

Appendices

doi:10.6342/NTU202404046

Appendix A

Detail of Implementation

Here we briefly describe the implementation of the protocol. The source code is organized by used entities, namely IA, VA, voter and tally. The main method/entrypoints are as follows.

A.1 IA

Source code for IA can be found at *IA/ia.py*. After initialize an IA instance, following methods are available.

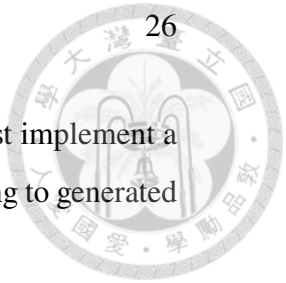
sign_ballot will accept two ballots and a identity, and return twp signed ballots after checking the identity. The method used to check identity must be implemented by the user.

get_rsa_public_key will return the RSA public key of the IA.

A.2 VA

Source code for VA can be found at *VA/va.py*. After initialize an VA instance, following methods are available.

doi:10.6342/NTU202404046



sign_ballot will accept a ballot and return a signed ballot. User must implement a method to check the validity of the ballot, whether by comparing to generated encrypted pairs or other algorithm.

get_rsa_public_key will return the RSA public key of the VA.

get_interactive_proof accept encrypted pairs and return a proof as described in the form of $((D, D')(D, D'))$.

get_commitment accept encrypted pairs and return a commitment as described.

generate_enc_pair return a list of encrypted pairs.

A.3 Voter

Source code for voter can be found at *voter/voter.py*. After initialize a voter instance, following methods are available.

receive_enc_pairs accept encrypted pairs and store it.

generate_ballot return two masked ballots as described.

A.4 Tally

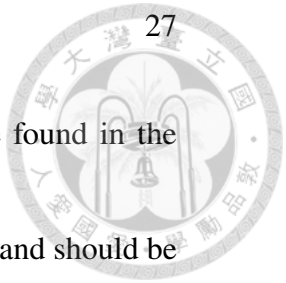
Contract source for tally can be found at *contract/main.mligo*, and a tally contract provides mainly following entrypoints.

init takes in IA and VA's RSA public key and public component of Benaloh system, then initialize the tally.

vote takes in voting value, mask and its inverse along with VA's signature and update tally state if all the value is correct.

finalize takes in decrypted result and proof and allows query for result.

doi:10.6342/NTU202404046



Deployer contract and complied version of both contract can be found in the *contract* directory too.

Notice that all implementation is not meant for production use, and should be used at own risk.

doi:10.6342/NTU202404046

Reference

[1] J. Benaloh and D. Tuinstra, "Receipt-free secret-ballot elections," in *Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*, 1994, pp. 544–553. 1, 3

[2] J. Benaloh, "Dense probabilistic encryption," in *Proceedings of the Workshop on Selected Areas of Cryptography*, 1994, pp. 120–128. 3, 10

[3] M. Hirt and K. Sako, "Efficient Receipt-Free Voting Based on Homomorphic Encryption," in *Advances in Cryptology — EUROCRYPT 2000*, B. Preneel, Ed. Berlin, Heidelberg: Springer, 2000, pp. 539–556. 3

[4] X. Chen, B. Lee, and K. Kim, "Receipt-Free Electronic Auction Schemes Using Homomorphic Encryption," in *Information Security and Cryptology - ICISC 2003*, J.-I. Lim and D.-H. Lee, Eds. Springer, 2004, pp. 259–273. 3

[5] A. Acquisti, *Receipt-Free Homomorphic Elections and Write-in Voter Verified Ballots*. Carnegie Mellon University, School of Computer Science [Institute for . . . , 2004. 3

[6] D. L. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Commun. ACM*, vol. 24, no. 2, pp. 84–90, 1981. 3

[7] T. Okamoto, "An electronic voting scheme," in *Advanced IT Tools: IFIP World Conference on IT Tools 2–6 September 1996, Canberra, Australia*, N. Terashima and E. Altman, Eds. Springer US, 1996, pp. 21–30. 4

doi:10.6342/NTU202404046

[8] A. Juels, D. Catalano, and M. Jakobsson, "Coercion-resistant electronic elections," in *Proceedings of the 2005 ACM Workshop on Privacy in the Electronic Society*. ACM, 2005, pp. 61–70. 4

[9] T. Moran and M. Naor, "Receipt-Free Universally-Verifiable Voting with Everlasting Privacy," in *Advances in Cryptology - CRYPTO 2006*, C. Dwork, Ed. Berlin, Heidelberg: Springer, 2006, pp. 373–392. 4, 19

[10] C. Killer, M. Eck, B. Rodrigues, J. von der Assen, R. Staubli, and B. Stiller, "ProvotuMN: Decentralized, Mix-Net-based, and Receipt-free Voting System," in *2022 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, 2022, pp. 1–9. 4

[11] A. Tariq, H. Binte Haq, and S. T. Ali, "Cerberus: A blockchain-based accreditation and degree verification system," *IEEE Transactions on Computational Social Systems*, vol. 10, no. 4, pp. 1503–1514, 2023. 4

[12] P. K. Rangi and P. Aithal, "A study on blockchain technology as a dominant feature to mitigate reputational risk for indian academic institutions and universities," *International Journal of Applied Engineering and Management Letters (IJAEML)*, vol. 4, no. 2, pp. 275–284, 2020. 4

[13] A. M. Al-madani, A. T. Gaikwad, V. Mahale, and Z. A. Ahmed, "Decentralized e-voting system based on smart contract by using blockchain technology," in *2020 International Conference on Smart Innovations in Design, Environment, Management, Planning and Computing (ICSIDEMPC)*, Oct. 2020, pp. 176–180. 4

[14] J. C. L. a. A. de Farias, A. Carniel, J. de Melo Bezerra, and C. M. Hirata, "Approach based on STPA extended with STRIDE and LINDDUN, and blockchain to develop a mission-critical e-voting system," *Journal of Information Security and Applications*, vol. 81, p. 103715, 2024. 4

doi:10.6342/NTU202404046

[15] T. Dimitriou, "Efficient, Coercion-free and Universally Verifiable Blockchain-based Voting," *Computer Networks*, vol. 174, p. 107234, 2020. 4, 5

[16] S. Gao, D. Zheng, R. Guo, C. Jing, and C. Hu, "An Anti-Quantum E-Voting Protocol in Blockchain With Audit Function," *IEEE Access*, vol. 7, pp. 115 304–115 316, 2019. 4

[17] F. Sheer Hardwick, A. Gioulis, R. Naeem Akram, and K. Markantonakis, "E-voting with blockchain: An e-voting protocol with decentralisation and voter privacy," in *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (Cpscom) and IEEE Smart Data (SmartData)*, 2018, pp. 1561–1567. 4

[18] C. Spadafora, R. Longo, M. Sala, and Department of Mathematics, University Of Trento, 38123 Povo, Trento, Italy, "A coercion-resistant blockchain-based E-voting protocol with receipts," *Advances in Mathematics of Communications*, vol. 17, no. 2, pp. 500–521, 2023. 4

[19] R. Ta¸s and O. O. Tanrıöver, "A Systematic Review of Challenges and Opportunities of Blockchain for E-Voting," *Symmetry*, vol. 12, no. 8, p. 1328, 2020. 4

[20] B. Wang, J. Sun, Y. He, D. Pang, and N. Lu, "Large-scale Election Based On Blockchain," *Procedia Computer Science*, vol. 129, pp. 234–237, 2018. 4

[21] Y. Wu, "An E-voting System based on Blockchain and Ring Signature," Master Thesis, University of Birmingham, 2017. 4

[22] W. Zhang, Y. Yuan, Y. Hu, S. Huang, S. Cao, A. Chopra, and S. Huang, "A Privacy-Preserving Voting Protocol on Blockchain," in *2018 IEEE 11th International Conference on Cloud Computing (CLOUD)*. IEEE, 2018, pp. 401–408. 4

doi:10.6342/NTU202404046

[23] L. V.-C. Thuy, K. Cao-Minh, C. Dang-Le-Bao, and T. A. Nguyen, "Votereum: An Ethereum-Based E-Voting System," in *2019 IEEE-RIVF International Conference on Computing and Communication Technologies (RIVF)*, 2019, pp. 1–6. 4

[24] B. Parno, J. Howell, C. Gentry, and M. Raykova, "Pinocchio: Nearly practical verifiable computation," *Commun. ACM*, vol. 59, no. 2, pp. 103–112, 2016. 5

[25] A. Emami, H. Yajam, M. A. Akhaee, and R. Asghari, "A scalable decentralized privacy-preserving e-voting system based on zero-knowledge off-chain computations," *Journal of Information Security and Applications*, vol. 79, p. 103645, 2023. 5

[26] A. M. Larriba, J. M. Sempere, and D. López, "A two authorities electronic vote scheme," *Computers & Security*, vol. 97, p. 101940, 2020. 7, 8

[27] A. M. Larriba and D. López, "A solidity implementation of tavs," *Frontiers in Blockchain*, vol. 6, p. 1105119, 2023. 7, 10

[28] L. Fousse, P. Lafourcade, and M. Alnuaimi, "Benaloh's Dense Probabilistic Encryption Revisited," in *Progress in Cryptology – AFRICACRYPT 2011*, A. Nitaj and D. Pointcheval, Eds. Springer, 2011, pp. 348–362. 10

doi:10.6342/NTU202404046