LaT-Voting: Traceable Anonymous E-Voting on Blockchain

Peng Li and Junzuo Lai(✉)

Jinan University, Guangzhou, China
penglijnu@gmail.com, laijunzuo@gmail.com

Abstract. In order to achieve anonymous voting, various cryptographic techniques are usually leveraged to avoid privacy leakage. However, traditional e-voting systems are excessively dependent on a centralized platform for publishing voting activity and storing ballots, which cannot ensure the security against tampering. Recently, numerous solutions based on blockchain are proposed to eliminate this threat. Nevertheless, these schemes are not applicable for wide adoption owing to the inefficiency in detecting double-voting. Meanwhile, there are no novel manners can both trace back to a malicious voter who voted twice, and discover his real identity simultaneously. Therefore, to settle the two aforementioned issues effectively, the first blockchain-based anonymous and traceable decentralized voting scheme called LaT-Voting is proposed in this paper. To better coordinate the conflict relationship between anonymity and accountability, we propose a new notion called *prefix-based linkable-and-traceable anonymous authentication*, which (i) achieves the authentication process without disclosing user privacy; (ii) provides a practical linkability to rapidly link two messages originated from a user; (iii) realizes a subtle traceability to track the user who authenticated twice.

Keywords: E-voting · Blockchain · Anonymous authentication · Linkability · Traceability

1 Introduction

Over the past decades, voting plays a critical role for people to exercise their power in modern society since its generation. As a problem-solving approach, it provides a straightforward and convenient way to make a decision according to a number of opinions especially when there are multiple choices. It ranges from boardroom voting, classroom voting and national election, etc. Traditional paper voting suffers from low efficiency, high cost and unintentional errors, thus it has gradually been replaced by electronic voting (e-voting) system which owns outstanding advantages and meets the development trend in modern society.

E-voting generally means the transmission and collection of people's suffrages by means of an electronic manner [1], which has obtained considerable concerns and interests as one of the most intractable cryptographic protocol problems [2].

J. K. Liu and X. Huang (Eds.): NSS 2019, LNCS 11928, pp. 234–254, 2019.
https://doi.org/10.1007/978-3-030-36938-5_14

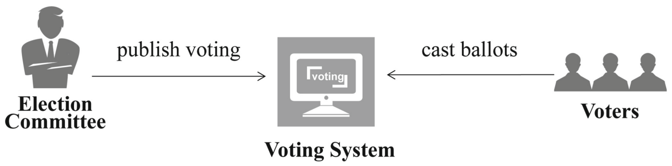

Fig. 1. Traditional e-voting system.

A typical e-voting model consists of three roles as illustrated in Fig. 1, i.e., an election committee, voters and a centralized voting system. An election committee publishes an election activity which usually includes multiple candidates through the centralized system. A large number of voters who have an eligible identity could cast ballots through the centralized voting system, while the election committee will then finish the ballot counting. A centralized voting system acts as a bulletin board to broadcast voting activity and collect all the ballots, which actually plays a role as a centralized database. Different cryptographic techniques have been proposed to establish e-voting systems, such as mix-net [3], blind signature [4], homomorphic encryption [5], group signature [6], ring signature [7], etc.

It is well known that reducing dependency of a centralized platform is desired in practice. However, in Fig. 1, a centralized platform unavoidably faces all the weaknesses of single point of failure. In contrast, the emerging blockchain technology provides a distributed, transparent and immutable ledger that ensures the transparency and reliability. The ledger is maintained by a group of nodes in a P2P network, and is verified by the whole network on the basis of a designated consensus protocol. This guarantees the realiable transmission through an untrusted network environment. More interestingly, the smart contract on top of blockchain can be automatically and securely executed, and will faithfully deal with all the message transmissions and perform related computations. Hence, blockchain can improve the transparency and ensure immutability in voting together with smart contract.

Despite of the limitation of the centralized model, e-voting schemes are subject to other unavoidable challenges. Firstly, traditional e-voting systems are so vulnerable to be tampered with, thus, it is naturally to be suspicious of the trustworthiness of the voting result. Secondly, users' sensitive information may be stored on the voting system together with the corresponding votes, which will raise the risk of privacy leakage. More importantly, double-voting is always unavoidable in most voting schemes, but there is no high-efficiency operations to check whether it happens. Last but not least, if a malicious voter casts twice, effective ways are needed to trace back to this voter without the cooperation of other parties.

There have been numerous researches to address these above mentioned challenges. Distributed architecture is designed to address single point of failure [8], but the blockchain technology not only realizes distributed storage but also maintains a immutable ledger to avoid data modification in a decentralized way.

Encryption schemes and signature mechanisms are generally applied to preserve voter's privacy. To check the double-casting while achieving anonymity, linkable group signature [6] and linkable ring signature [9] can be used as tools to link any two ballots sent by an identical voter, but they need considerable exponent calculations which are inefficient to implement linkability quickly. Furthermore, to find out who casts twice, Tracing-by-Linking group signature [10], linkable threshold ring signature [7], traceable ring signature [11] can realize the function of traceability. However, these related implementations are impractical since these schemes involve complicated constructions and massive computations. In order to track a misbehaving voter, the group manager in the general group signature schemes is capable of opening all members' signatures [12] while increasing the risk of privacy breach. Other related threshold schemes in [13] and [14] can trace a ballot and its owner, but they need to introduce additional trusted parties to join in.

Until now, none of the existing researches has resolved all of the above mentioned issues simultaneously. Therefore, our work is motivated to design and achieve a decentralized e-voting with security, anonimity, linkability, traceability in a concise and efficient manner. To fulfill this task, we have to address three challenges: design a rapid method to link two ballots voted by the same voter, coordinate the conflict relationship between anonymity and accountability. Accordingly, we investigate an effective manner for not only rapidly detecting double-casting, but also grasping the double-voted voter and exposing his identity. To achieve the rapid linkability, we choose the innovative trick of common-prefix in [15] to reach this goal. We also realize the public traceability based on [16] so as to further reveal the double-voted voter's identity without any trusted authority.

Our Contributions. In this paper, we propose the blockchain-based anonymous and traceable decentralized voting scheme called LaT-Voting, which ensures the ballots confidentiality and voter anonymity while achieving stronger accountability. Specifically,

(1) We design and construct a blockchain-based decentralized e-voting scheme, which is not dependent on any central third party to complete voting process. It ensures the security of ballot content by supporting a distributed way to store encrypted ballots. Moreover, the immutable ledger of blockchain acts as a secure and tamper-resistant database so that to avoid data modification.
(2) To achieve the speedy linkability while better coordinating the anonymity and accountability, we propose a concrete scheme called prefix-based linkable-and-traceable anonymous authentication, which realizes authentication without any privacy disclosing, and provides a practical linkability to rapidly link two authenticated messages sent from an identical user. More importantly, the public traceability is implemented to track the misbehaving user who authenticated twice or more. Particularly, it is anonymous if a user authenticates once, since anyone cannot tell one's identity from the authentication transcript. However, the linkability and public traceability

are activated as long as two messages authenticated by a same key, everyone can link the two messages and expose the user's identity.

(3) We design the voting scheme on top of blockchain, and illustrate the voting process by smart contract. Smart contract is used for collecting ballots and verifying their validity, and exposing the double-voted voters. In addition, we provide the security analysis of our proposed voting scheme.

The remainder of the paper is organized as follows. The related work is present in Sect. 2, and in Sect. 3 we will give the essential preliminaries. We describe the system model in Sect. 4, and propose the concrete voting scheme together with the linkable-and-traceable anonymous authentication in Sect. 5. Next, we will give the correctness and security of LaT-Voting in Sect. 6, and analysis the evaluation results in Sect. 7. Finally, we summarize our research work of this paper in Sect. 8.

2 Related Work

Researching on e-voting has gained considerable attention and interest with the swift development of the Internet. We review some representative works.

Centralized Voting. Numerous e-voting systems are usually carried out in a centralized manner. The essential processes of voting, like ballots collection and ballots tallying, are executed and managed by the role of authority, such as the system administrator or the group manager, which may make the privacy in danger. The centralized services are involved in the execution of voting protocol in [17] and [18], but [17] can keep voters in anonymity and [18] distributes trust to different authorities. In addition, a centralized platform is normally used to store voting data, which is easily attacked by malicious adversaries and will be subject to the trouble of single point of failure and privacy disclosure. Helios in [19] and the protocol in [20] are both implemented with a web-based bulletin board that enables anyone to observe the dynamic process in the voting, but it usually suffers from privacy disclosure and the awkward issue of single point of failure.

Distributed Voting. There are several studies on the distributed voting schemes. [21] considers a distributed asynchronous system to realize the ballot collection in every server. [22] introduces a distributed architecture for web-based voting, which distributes the task of collecting ballots over multiple servers to reduce the opportunity of the single failure in tallying. [23] designs a distributed architecture for allowing to vote at any voting station, [24] implements a distributed processing architecture to process ballots over several web servers. [25] presents a distributed voting system with a fully asynchronous ballot collection subsystem which includes a number of nodes to provide immediate assurance that the ballot is recorded as the voter cast. Although the researches in [21–25] aim to convert the treatment to a distributed fashion in e-voting, their schemes are actually dependent on a centralized system to offer services, which is inconsistent with our purpose to devise a decentralized e-voting protocol.

Decentralized Voting. There are also various researches concentrated on designing voting protocols without relying the role of authority or a central platform. The typical self-tallying (such as [2,26,27]) protocol achieves that the process of voting is carried out by the voters themselves without the involvement of other parties, which provides stronger privacy protection and supports the dispute-freeness. But the drawback of this kind of protocol is that it requires all the voters must join in the voting. Unfortunately, even a single one that deviates from the protocol will result in the failure of tallying. Therefore, it is not suitable for large scale voting due to its non-scalability.

Blockchain-Based Voting. [28] realizes a self-tallying voting using Bitcoin, the ballots are not required to be encrypted, but with the help of a commitment process which chooses a randomness to hide the real value of the ballot and needs the sum of all the random numbers is zero. [29] also implements a self-tallying voting on blockchain through designing a two-round protocol. However, [28,29] only support two candidates, even though they are decentralized via taking advantage of the self-tallying. [30] is a decentralized voting which supports self-tallying and multiple candidates. In addition, [31] establishes a platform-independent voting system to eliminate the limitations on the number of voters and candidates, [32] attempts to remove the role of any trusted party by using blockchain, but both of them only make the ballots to be secret. The above mentioned researches are limited to their specific requirements, but our work focus on holding the anonymity while keeping the accountability.

Traceable Voting. In order to preserve privacy, typical e-voting systems usually reveal the feature of untraceability [21,33–35] that separates the relevancy between the ballot and its owner. Mixnet [35], DC-net [33,36], blind signature [37] and ring signature [38] make it possible to attain untraceability, but the double-casting becomes an intractable issue needs to be worked out. It is necessary to balance privacy and accountability so as to prevent from abusing the anonymity. Linkable group signature in [6] and linkable ring signature in [1,9] have been suggested to be used in constructing e-voting system, but it can only verify whether two valid signatures are from the same signer. However, the two kinds of signature schemes make linking-and-tracing come true in [10] and [7]. But both the ring signature and group signature schemes are not succinct to reach the goal of traceability. Similarly, traceable ring signature [11] could be applied to voting, which realizes the public traceability that can track the public key of the owner who signed two signatures, but massive computation and complex operations are involved. In addition, [13] introduces revocable anonymity to voting, which can recover the identity of the voter, yet it must rely on the coordination with other parities. [14] constructs a complex voting protocol which satisfies the anonymity and traceability simultaneously, but only allowing the administrator to track the ballot and locate the voter with the help of other trusted parties. The above mentioned studies are constrained to accomplish the public traceability while holding the anonymity. By comparison, our scheme provides a subtle protocol with anonymity and public traceability in e-voting for the purpose of avoiding anonymity misuse and holding accountability.

3 Preliminaries

3.1 Blockchain

Blockchain is a distributed, transparent and immutable ledger that consists of a large number of transactions which are recorded in blocks. Each block involves a certain number of transactions and a hash of the previous block, and is generated according to a predefined consensus protocol. Numerous blocks in sequence connect to a chain which we called blockchain. Blockchain offers a distributed manner to keep a consistent replicate and avoid data tampering by rewarding miners for jointly maintaining the blockchain network.

In general, blockchain can be regarded as a public ledger. The data that will be recorded on the blockchain is sent to the blockchain network in the form of a transaction, which is signed and sent by the user. After the transaction is broadcast to the whole network, it will be verified and packed into a block together with other transactions, these transactions will be written into the blockchain after the block is confirmed. On the other hand, it is visible to view all the data transmissions for the public, thus, anyone is available to check the validity of all the transactions.

A blockchain address is a randomly generated anonymous address by computing a hash of the public key. Thus, the address acts as a pseudoym to hide user's identity. A transaction should be signed by the corresponding secret key, no one can send a transaction through a random address without the corresponding secret key.

3.2 Smart Contract

A smart contract can be considered as a self-executing computer program [29] which transfers the running processes into executable codes as a legal agreement. The necessary fairness and credibility can be ensured directly through the automated execution of contract code. The reason that it cannot be widely applied to represent its intelligence before is mainly due to the lack of a secure and decentralized development environment [8]. However, in the blockchain distributed network, each node updates the duplicate locally based on the current execution result after runing the smart contract. Blockchain has the possibility of providing a trustworthy and decentralized mode to accelerate its exploitation. Currently, smart contracts are designed as multiple kinds of reliable and decentralized applications while offering fair and secure services to avoid dishonesty, downtime and tampering.

3.3 zk-SNARK

Zero-knowledge proof is a delicate cryptographic protocol, which enables a prover to generate a proof and convince the verifier that he indeed knows a secret without leaking any additional knowledge of the secret. In voting protocols, zero-knowledge proof is widely employed to guarantee the operations are not deviated from the required rules. In this paper, we exploit the zk-SNARK [39] to accomplish the specific proof for a NP-language $\mathcal{L} = \{x|\exists w, s.t., C(x, w) = 1\}$ [15], where x is a statement, w is a witness for x, and C is a boolean circuit. Informally, zk-SNARK is made of three algorithms:

- $\mathsf{KeyGen}(1^\lambda, C) \longrightarrow (pk, vk)$. The KeyGen algorithm inputs a security parameter 1^λ and a circuit C, and outputs a proving key pk and a verification key vk. Both of the keys are public parameters.
- $\mathsf{Prover}(pk, x, w) \longrightarrow \pi$. The Prover algorithm inputs the proving key pk, a statement x, and a witness w. It outputs a proof π.
- $\mathsf{Verifier}(vk, x, \pi) \longrightarrow 0/1$. The $\mathsf{Verifier}$ algorithm inputs the verification key vk, the statement x, and the corresponding proof π. It outputs 1 if π is a valid attestation for $x \in \mathcal{L}$.

Informally, the zk-SNARK satisfies the following properties.

- *Completeness.* A prover can generate a proof such that it can be passed through the verification by the verifier with probability 1.
- *Soundness.* No polynomial-time adversary is capable of forging a valid attestation that can be accepted by the verifier with non-negligible probability.
- *Efficiency.* The randomized algorithms run in time polynomial in the sizes of the corresponding input.
- *Zero-knowledge.* The procedure only reveals the statement rather than any secret.
- *Proof of knowledge.* If the verifier accepts a statement from a prover, there is a polynomial-time extractor who can generate a valid witness when giving oracle access to the prover.

4 System Model

4.1 Entities

As shown in Fig. 2, four entities are involved in the voting protocol, namely, the certificate authority, the election committee, voter and the smart contract.

- *Certificate Authority*, identified by CA, is mainly to certify and manage the user's identity, and issues a related certificate to the user.
- *Election Committee*, identified by EC, is the organizer of voting, whose task is to post a voting activity and compute the election result.
- *Voter*, identified by V_i, is the participant who has the right to cast a ballot before deadline.
- *Smart Contract*, identified by SC, is designed primarily for collecting and verifying ballots, detecting double-voting and tracing the double-voted voters.

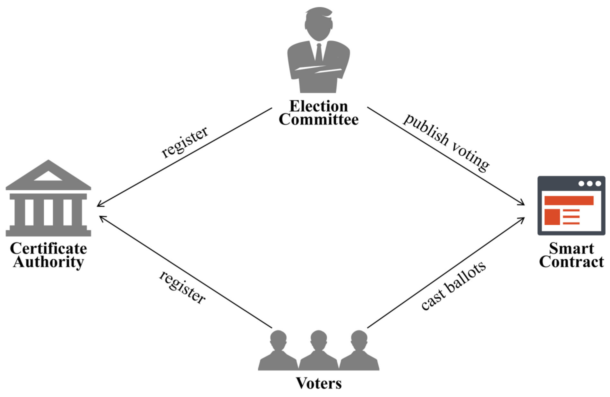

Fig. 2. The system model of LaT-Voting.

4.2 Requirements

Our LaT-Voting scheme requires the following properties.

- *Privacy:* The content of ballot is kept secret from the public, except the role who performs the tallying.
- *Anonymity:* The voter should be kept anonymous when casting a ballot, and no one can link a ballot to a voter.
- *Unforgeability:* No one can forge user's identity to generate a ballot, and anyone can test whether a ballot is generated correctly.
- *Public verifiability:* Any party should be convinced that all ballots have been counted when tallying, and the final election result can be verified.
- *Traceability:* Double-voted behavior can be detected by seeking the linkability of two ballots. Any voter who voted twice will be discovered and his identity will be tracked and exposed.

5 Traceable Anonymous E-Voting

In this section, we will firstly present the new notion, called prefix-based linkable-and-traceable anonymous authentication, to support an anonymous yet accountable requirement, and then describe how our decentralized e-voting scheme works so as to ensure security, anonymity, linkability and traceability. The voting protocol is tested on an open blockchain.

5.1 Prefix-Based Linkable-and-Traceable Anonymous Authentication

The prefix-based linkable-and-traceable anonymous authentication can be built on any certificate generation procedure, and therefore we can construct it as other existing schemes. Meanwhile, we prefer to design a non-interactive authentication which can be represented as algorithms rather than protocols.

Syntax. Formally, a prefix-based linkable-and-traceable anonymous authentication (PLTAA) scheme consists of the following six algorithms.

- $\mathsf{Setup}(1^\lambda) \longrightarrow (\mathsf{MPK}, \mathsf{MSK})$. The setup algorithm is a function that takes as input a security parameter 1^λ, and outputs a master public key MPK and a master secret key MSK.
- $\mathsf{CertGen}(pk, \mathsf{MSK}) \longrightarrow \sigma$. The certificate generation algorithm takes as input a user's public key pk and the master secret key *MSK*. It outputs a certificate σ that can validate the corresponding public key pk.
- $\mathsf{Auth}(m = m_l || m_r, pk, sk, \sigma, \mathsf{MPK}) \longrightarrow \pi$. The authentication algorithm takes as input a message m that has a left part m_l (i.e. a prefix) and a right part m_r, the user's public key pk and secret key sk, the certificate σ, and the master public key MPK. It outputs an authentication token π on the message m to show that the user who created the message m indeed has the secret key corresponding to a valid certificate.
- $\mathsf{Verify}(m, \pi, \mathsf{MPK}) \longrightarrow 1/0$. The verification algorithm takes as input a message m, an authentication token π and the master public key MPK. It outputs 1 or 0 to decide whether the proof is valid or not.
- $\mathsf{Link}(m_1, m_2, \pi_1, \pi_2) \longrightarrow 1/0$. The link algorithm takes as input two messages m_1, m_2 and their corresponding authentication tokens π_1, π_2. It outputs 1 if m_1 and m_2 have a common prefix with a fixed length; otherwise, it outputs 0.
- $\mathsf{Trace}(\pi_1, \pi_2) \longrightarrow pk$. The trace algorithm takes as input the two authentication tokens π_1, π_2 corresponding to the linked two messages. It outputs the traced public key pk, which points to the user who authenticates two messages having a common prefix.

Correctness. Correctness of PLTAA scheme must satisfy:

- *Verification correctness.* Authentication tokens generated according to specification will successfully pass the verification by Verify.
- *Linking correctness.* If two messages sharing a common prefix are authenticated by a user, they can certainly be linked by Link.
- *Tracing correctness.* If two messages are linked, the same originator who authenticated them will be traced by Trace.

Notions of Security. Security of PLTAA schemes has four aspects: unforgeability, anonymity, linkability and accountability.

Unforgeability. Unforgeability for PLTAA schemes is defined as the following game between the Challenger $\mathcal{C}$ and the Adversary $\mathcal{A}$.

1. $\mathcal{A}$ creates a master key pair $(\mathsf{MPK}, \mathsf{MSK})$.
2. $\mathcal{A}$ performs the certificate generation process with $\mathcal{C}$. When $\mathcal{C}$ submits a public keys pk to $\mathcal{A}$, $\mathcal{A}$ returns a certificates σ to $\mathcal{C}$.

3. $\mathcal{A}$ randomly chooses q message $m_1, \ldots, m_q$ and asks $\mathcal{C}$ to authenticate them. $\mathcal{C}$ produces and outputs the corresponding authentication tokens $\pi_1, \ldots, \pi_q$.
4. $\mathcal{A}$ selects a message m and creates the corresponding π, and outputs (m, π).

$\mathcal{A}$ wins if:

1. $\mathsf{Verify}(m, \pi, \mathsf{MPK}) = 1$;
2. $m \in \{m_1, \ldots, m_q\}$.

We denote by

$$\mathbf{Adv}_{\mathcal{A}}^{Unfo}(\lambda) = \mathsf{Pr}[\mathcal{A}\ wins\ the\ game].$$

Definition 1 (Unforgeability). *A* PLTAA *scheme is unforgeable, if for all PPT adversary* $\mathcal{A}$*,* $\boldsymbol{Adv}_{\mathcal{A}}^{Unfo}(\lambda)$ *is negligible.*

Anonymity. Anonymity for PLTAA schemes is defined as the following game between the Challenger $\mathcal{C}$ and the Adversary $\mathcal{A}$.

1. $\mathcal{A}$ creates a master key pair $(\mathsf{MPK}, \mathsf{MSK})$.
2. $\mathcal{A}$ performs the certificate generation process with $\mathcal{C}$. When $\mathcal{C}$ submits two public keys pk_1, pk_2 to $\mathcal{A}$, $\mathcal{A}$ returns the corresponding certificates σ_1, σ_2 to $\mathcal{C}$.
3. $\mathcal{A}$ chooses a messages m and asks $\mathcal{C}$ to authenticate it. $\mathcal{C}$ randomly picks $b \in \{1, 2\}$, uses (pk_b, sk_b, σ_b) to authenticate m, and sends the newly generated authentication token π_b to $\mathcal{A}$.
4. After receiving π_b, $\mathcal{A}$ outputs the guess b'.

$\mathcal{A}$ wins if $b' = b$. We denote by

$$\mathbf{Adv}_{\mathcal{A}}^{Anon}(\lambda) = \left| \mathsf{Pr}[\mathcal{A}\ wins\ the\ game] - \frac{1}{2} \right|.$$

Definition 2 (Anonymity). *A* PLTAA *scheme is anonymous, if for all PPT adversary* $\mathcal{A}$*,* $\boldsymbol{Adv}_{\mathcal{A}}^{Anon}(\lambda)$ *is negligible.*

Linkability. Linkability for PLTAA schemes is defined as the following game between the Challenger $\mathcal{C}$ and the Adversary $\mathcal{A}$.

1. $\mathcal{C}$ creates a master key pair (MPK, MSK) and gives $\mathcal{A}$ the master public key MPK.
2. $\mathcal{C}$ performs the certificate generation process with $\mathcal{A}$. When $\mathcal{A}$ submits a public key pk to $\mathcal{C}$, $\mathcal{C}$ returns a certificate σ to $\mathcal{A}$.
3. $\mathcal{C}$ chooses two messages $m_l || m_1, m_l || m_2$ sharing a common prefix m_l to $\mathcal{A}$, and asks $\mathcal{A}$ to authenticate them. $\mathcal{A}$ creates the corresponding authentication tokens π_1, π_2, and outputs $(m_l || m_1, \pi_1), (m_l || m_2, \pi_2)$.

$\mathcal{A}$ wins if:

1. $\mathsf{Verify}(m_l||m_i, \pi_i, \mathsf{MPK}) = 1$ for $i = 1, 2$;
2. $\mathsf{Link}(m_l||m_1, m_l||m_2, \pi_1, \pi_2) = 0$.

We denote by

$$\mathbf{Adv}_{\mathcal{A}}^{Link}(\lambda) = \mathsf{Pr}[\mathcal{A} \textit{ wins the game}].$$

Definition 3 (Linkability). *A PLTAA scheme is linkable, if for all PPT adversary $\mathcal{A}$, $\boldsymbol{Adv}_{\mathcal{A}}^{Link}(\lambda)$ is negligible.*

Accountability. Accountability for PLTAA schemes is defined as the following game between the Challenger $\mathcal{C}$ and the Adversary $\mathcal{A}$.

1. $\mathcal{C}$ creates a master key pair (MPK, MSK) and gives $\mathcal{A}$ the master public key MPK.
2. $\mathcal{C}$ performs the certificate generation process with $\mathcal{A}$. When $\mathcal{A}$ submits a public key pk to $\mathcal{C}$, $\mathcal{C}$ returns a certificate σ to $\mathcal{A}$.
3. $\mathcal{C}$ chooses two messages $m_l||m_1, m_l||m_2$ sharing a common prefix m_l to $\mathcal{A}$, and asks $\mathcal{A}$ to authenticate them. $\mathcal{A}$ creates the corresponding authentication tokens π_1, π_2, and outputs $(m_l||m_1, \pi_1), (m_l||m_2, \pi_2)$.

$\mathcal{A}$ wins if:

1. $\mathsf{Verify}(m_l||m_i, \pi_i, \mathsf{MPK}) = 1$ for $i = 1, 2$;
2. $\mathsf{Link}(m_l||m_1, m_l||m_2, \pi_1, \pi_2) = 1$;
3. $\mathsf{Trace}(\pi_1, \pi_2) = pk'$, but $pk' \neq pk$; or, $\mathsf{Trace}(\pi_1, \pi_2) = \perp$.

We denote by

$$\mathbf{Adv}_{\mathcal{A}}^{Acco}(\lambda) = \mathsf{Pr}[\mathcal{A} \textit{ wins the game}].$$

Definition 4 (Accountability). *A PLTAA scheme is accountable, if for all PPT adversary $\mathcal{A}$, $\boldsymbol{Adv}_{\mathcal{A}}^{Acco}(\lambda)$ is negligible.*

Construction. We proceed to construct such a PLTAA scheme. Similar to other anonymous authentication schemes, we also utilize the zero-knowledge proof technique to achieve anonymity. Specifically, we apply zk-SNARK to obtain an efficient construction. Since the assurance contributing to linkability-and-traceability is the prefix, consequently, we will exploit it to support an anonymous-yet-accountable requirement. In a nutshell, the authentication process creates a linking tag committed to the prefix and the user's secret key. To satisfy the requirement of accountability, we will also provide a tracing tag which will be used for tracking.

Let $\mathsf{S} = (\mathsf{S.Setup}, \mathsf{S.Sign}, \mathsf{S.Verify})$ be a signature scheme, $\mathsf{Z} = (\mathsf{Z.Setup}, \mathsf{Z.Prover}, \mathsf{Z.Verifier})$ be the zk-SNARK protocol. Also, assume $\mathcal{H} : \{0,1\}^* \longrightarrow \{0,1\}^n$ represents a secure hash function, the public function $\mathsf{F} : pk = \mathsf{F}(sk)$ denotes a key-pair verification algorithm. The PLTAA scheme is as follows.

- $\mathsf{Setup}(1^\lambda)$. The setup algorithm first invokes $\mathsf{S.Setup}(1^\lambda)$ to create a signing key msk and a verification key mpk, and calls $\mathsf{Z.Setup}(1^\lambda)$ to obtain the public parameters PP for zk-SNARK. The public parameters are $\mathsf{MPK} = (\mathcal{H}, mpk, \mathsf{PP})$, the master secret key is $\mathsf{MSK} = (msk)$.
- $\mathsf{CertGen}(pk_i, \mathsf{MSK})$. The certificate generation algorithm calls $\mathsf{S.Sign}(m, msk)$ to compute a signature σ_i on a public key pk_i, and outputs σ_i.
- $\mathsf{Auth}(p||m_r, pk_i, sk_i, \sigma_i, \mathsf{MPK})$. On input a message $p||m_r$ including a prefix p, the authentication algorithm dose the following:
  1. Compute a linking tag $t_1 = \mathcal{H}(p, sk_i)$ and a tracing tag $t_2 = \mathcal{H}(p, pk_i, sk_i) + m_r \cdot sk_i$.
  2. Let $x = (p||m_r, \mathsf{MPK})$ be the common knowledge, $w = (pk_i, sk_i, \sigma_i)$ be the private witness. Call key-pair verification algorithm $pk = \mathsf{F}(sk)$, certificate verification algorithm $\mathsf{S.Verify}(mpk, m, \sigma)$ for the language

$$\begin{aligned}\mathcal{L} = \{&t_1, t_2, x = (p||m_r, \mathsf{MPK})|\ \exists w = (pk_i, sk_i, \sigma_i), s.t.,\\ &pk_i = \mathsf{F}(sk_i) \wedge \mathsf{S.Verify}(pk_i, \sigma_i, mpk) = 1 \wedge\\ &t_1 = \mathcal{H}(p, sk_i) \wedge t_2 = \mathcal{H}(p, pk_i, sk_i) + m_r \cdot sk_i\},\end{aligned}$$

     where the function F is to confirm if the two keys correspond to a public-secret key pair, the $\mathsf{S.Verify}$ algorithm is used for checking whether σ_i is a valid certificate. Next, call zk-SNARK proving algorithm $\mathsf{Z.Prover}(x, w, \mathsf{PP})$ to produce a proof η related to the statement $x \in \mathcal{L}$.
  3. At last, the algorithm outputs an authentication token $\pi = (t_1, t_2, \eta)$.
- $\mathsf{Verify}(p||m_r, \pi, \mathsf{MPK})$. The verification algorithm invokes $\mathsf{Z.Verifier}(x, \pi, \mathsf{PP})$, and outputs 0 or 1 for invalid or valid, respectively.
- $\mathsf{Link}(m_1, m_2, \pi_1, \pi_2)$. Let $\pi_1 = (t_1^1, t_2^1, \eta_1)$, $\pi_2 = (t_1^2, t_2^2, \eta_2)$, the link algorithm checks whether t_1^1 in π_1 equals to t_1^2 in π_2. If $t_1^1 = t_1^2$, it outputs 1 for linked; otherwise, it outputs 0.
- $\mathsf{Trace}(\pi_1, \pi_2)$. If $t_1^1 = t_1^2$, the trace algorithm computes a derived secret key sk_i and calls the function F to calculate the corresponding public key $pk_i = \mathsf{F}(sk_i)$ pointing to the user i.

The correctness and security theorems are given in Appendix A.

5.2 The LaT-Voting Protocol

Now we prepare to describe a general protocol for a type of voting scheme with traceability back to malicious voters who voted twice or more in a voting activity. Notice that we let each user generate a distinct blockchain address for a voting activity, namely, each user has a unique address, which is regarded as a simple solution to achieve anonymity in the blockchain network. And on the basis of this protocol, we can also extend it to additional voting schemes by appending an incentive mechanism to reward honest participants.

The workflow of LaT-Voting is depicted as follows.

(1) Each participant should register at CA to obtain a certificate. Then, they can post a voting activity or cast a ballot.

(2) *EC* prepares to post a voting activity. After the voting is announced, each voter is allowed to join in.
(3) When receiving the notification of voting, each voter can cast a ballot before deadline.
(4) *SC* verifies every ballot and picks out all double-voted ballots. Then *EC* calculates the election result according to the valid ballots.
(5) *SC* tracks the identities of double-voted voters according to the linked ballots.

The whole process of our LaT-Voting protocol is shown in Fig. 3, more details are described as follows.

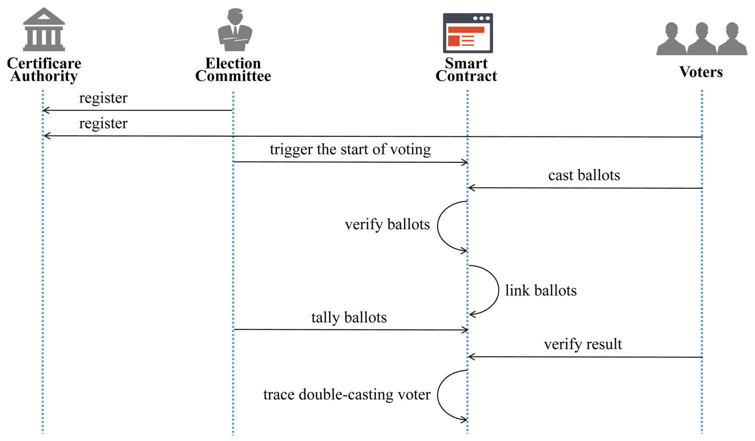

Fig. 3. The process of LaT-Voting.

Register. *CA* calls S.Setup to create the signing key pair (mpk, msk). *EC* generates his key pair (pk_{EC}, sk_{EC}), and registers at *CA* to get a certificate σ_{EC}. Similarly, V_i also creates his key pair (pk_i, sk_i), and registers at *CA* to get a certificate σ_i. This registration process is normally done offline only once for each user by calling S.Sign algorithm.

Publish. Before starting a voting activity, *EC* should create a new blockchain account address α_{EC}, a key pair (epk, esk) that is only for this activity, and the epk will be used for voters to encrypt their own votes. *EC* prepares the related parameters Param containing a unique voting activity serial number Num, a list of k candidates (each one corresponds to a unique identifier ID_j), the public parameters MPK (i.e. a hash function $\mathcal{H}$, *CA*'s public key mpk, the SNARK's public parameters PP), the encryption key epk, the deadline t, the tally mechanism T, and an authentication token π_{EC} on Num and α_{EC}. π_{EC} can be created through calling Auth algorithm, i.e.,

$$\pi_{EC} = \mathsf{Auth}(Num||\alpha_{EC}, pk_{EC}, sk_{EC}, \sigma_{EC}, \mathsf{MPK}).$$

Then EC codes a voting smart contract SC which contains all above public parameters for this voting activity. After successfully compiling the contract SC, he puts the code of SC into a transaction, and sends the newly constructed transaction into the blockchain network through the previously generated address α_{EC}.

Cast. Once a new generated block includes the above transaction in blockchain, everyone could retrieve the transaction and the contract SC. Voters first check if the address α_{EC} corresponds to the one in contract SC, and test the validity of contract SC. Then, V_i starts to create his one-time anonymous blockchain address α_i, then encrypts his ballot with the public key epk to produce a corresponding encrypted ballot C_i.

Additionally, V_i invokes Auth algorithm to attach an authentication token

$$\pi_i = \mathsf{Auth}(Num||C_i, pk_i, sk_i, \sigma_i, \mathsf{MPK}).$$

Finally, V_i puts (C_i, π_i) into a transaction and sends them to the blockchain network using the one-time address α_i before the deadline.

Tally. SC runs $\mathsf{Verify}(Num||C_i, \pi_i, \mathsf{MPK})$ to check the validity of π_i. If π_i is valid, SC will proceed to run $\mathsf{Link}(C_i, C_*, \pi_i, \pi_*)$ for each ballot and its authentication token. C_* is the verified ballot which has passed through the Verify algorithm, π_* is the corresponding authentication token. This operation ensures that C_i is the first and valid submission from an eligible voter. Yet for any unauthenticated or double-voted ballot, the contract will drop it.

Finally, SC picks out all the valid votes till the deadline. Then EC will retrieve them, encrypt and calculate the election result $r = (r_1, \ldots, r_k)$. EC publishes r together with a corresponding zero knowledge proof π_{result} to prove the process is properly executed. Concretely, π_{result} is for the NP-language

$$\begin{aligned}\mathcal{L}_r = \{\mathsf{Param}, C^*, r|\ \exists esk, s.t., epk = \mathsf{F}(esk) \wedge B_i = \mathsf{Dec}(esk, C_i)\\ \wedge_{j=1}^{k} r_j = \mathsf{T}(ID_j; B_1, \ldots, B_n)\},\end{aligned}$$

where C^* contains all the valid ballots in tallying, B_i indicates the plaintext of the encrypted ballot C_i, r represents the final voting result for k candidates, the Dec algorithm is to decrypt each encrypted ballot, r_j is the sum of ballots of each candidate, T denotes a tallying mechanism for all the candidates. And the secret key esk acts as the witness to establish an effective attestation.

At last, EC puts the result r and the corresponding proof π_{result} into a transaction, and sends it to the blockchain network using his address α_{EC}. After a new block containing r and π_{result} is appended to the blockchain, each one can verify the correctness of the election result.

Trace. If a voter casts two ballots C_i and C_i', there will be two authenticated tokens π_i and π_i' containing two distinct tracing tags. Specifically,

$$t_2 = \mathcal{H}(Num, pk_i, sk_i) + C_i \cdot sk_i,$$

$$t_2' = \mathcal{H}(Num, pk_i, sk_i) + C_i' \cdot sk_i.$$

Since the encrypted vote is randomly generated even though it encrypts a same value, thus C_i and $C_i^{'}$ are different. Consequently, the secret key of the double-voted voter can be easily derived by calling Trace algorithm, i.e.

$$sk_i = \frac{t_2 - t_2'}{C_i - C_i'}.$$

Then SC can locate this voter according to the public key $pk_i = \mathsf{F}(sk_i)$.

6 Correctness and Security Analysis

6.1 Correctness Analysis

The correctness is satisfied by the public verifiability which is provided by the cryptographic schemes and the blockchain platform. The cryptographic schemes achieve: (i) the certificate generated by S.Sign can be verified by S.Verify, (ii) the key pair can be verified by the function F; (iii) the proof created by Z.Prover can be verified by Z.Verifier; (iv) the linking-and-tracing correctness can be ensured by our PLTAA scheme. In addition, the blockchain platform serves as a decentralized public bulletin which contains the overall ballots from all the qualified voters. Meanwhile, the blockchain also ensures the consistent execution of all the transactions, which makes all the mining nodes mutually maintain an identical distributed ledger that contains the overall transactions mentioned above. And any inconsistency that causes an error would lead to the rejection of the transaction. It means that all the voters can be convinced that each ballot on the blockchain will be checked and verified by all the mining nodes, and any invalid ballot in the transaction will be refused to be appended to the ledger unless an adversary takes control of a certain percentage of compromised nodes in the blockchain network.

6.2 Security Features of the LaT-Voting Protocol

- *Privacy:* The ballots stored on the blockchain are in the form of ciphertext, which are encrypted by the key epk, and only the corresponding key esk could decrypt them. It realizes the ballot confidentiality that only EC with the esk other than anyone can calculate the final election result, and the ballot information is unknown to other parities.
- *Anonymity:* Every voter casts his ballot using a randomly generated one-time address α_i which prevents an adversary from linking a voter through his multiple addresses that have ever been involved in various transactions. Meantime, each authentication token $\pi_i = (t_1, t_2, \eta)$ created by Auth has two hash values $\mathcal{H}(Num, sk_i)$, $\mathcal{H}(Num, pk_i, sk_i)$ that can be considered as random numbers without revealing voter's any information, which realizes the anonymity based on our PLTAA scheme.

- *Unforgeability:* No one can forge a linking tag $t_1 = \mathcal{H}(Num, sk_i)$ in his authentication token such that it will be linked with another linking tag which is not generated by him. That is to say, an adversary cannot fake other voters' identities to cast a ballot. It is guaranteed by the unforgeability feature of our PLTAA scheme. Consequently, an adversary cannot forge a ballot in the name of a voter or fabricate an authentication token to track the voter's identity.
- *Public verifiability:* The whole process of voting is presented as a series of transactions, and stored as multiple blocks in a distributed ledger. Due to the openness and verifiability of the blockchain platform, the public have the right to verify whether all the ballots are recorded on the blockchain and the related authentication token are correctly generated, and check the validity of the final election result. In other words, it ensures the truthfulness in a transparent and verifiable fashion.
- *Traceability:* We fully capitalize on the PLTAA scheme so as to establish the linkability in a novel manner. It can rapidly detect whether there exists two messages authenticated by a same voter according to $\mathcal{H}(Num, sk_i)$, thus it is impossible for an adversary to authenticate two messages without being linked. Besides only linking the two ballots when double casting happens, on the contrary, we further try to infer his real identity based on the submitted information, even though he did that anonymously. Therefore, PLTAA scheme also provides an innovative strategy for tracking back to the double-casting voter who indeed submitted two authenticated ballots according to the tracing tag $t_2 = H(Num, pk_i, sk_i) + C_i \cdot sk_i$. Hence, as long as any two valid ballots corresponding to an identical voter, our solution could make his identity to be forcibly disclosed to the public in an ingenious way so as to enable the double voting can be avoided.

7 Conclusion

We present the LaT-Voting, a blockchain-based anonymous and traceable decentralized voting protocol, which provides novel privacy protection against disclosing personal identity information, and a practical linkability to rapidly link two ballots sent from the same voter. Moreover, to relieve the tension between anonymity and accountability, we achieve a subtle traceability to track the participant who is double-voted. We put forward a new notion called linkable-and-traceable anonymous authentication to meet the anonymous yet accountability requirement. A specific construction of this scheme is proposed, and it shows the usability and compatibility to the voting protocol and the blockchain infrastructure. We further list the security features of our voting protocol.

A Correctness and Security Theorems

We outline the correctness and security theorems for the construction of our PLTAA scheme.

Correctness. Given a key pair (pk, sk) and a prefix p, we assume that the certificate σ is the output of $\mathsf{S.Sign}(sk, \mathsf{MPK})$, the authentication token $\pi = (t_1, t_2, \eta)$ is the output of $\mathsf{Auth}(p||m, pk, sk, \sigma, \mathsf{MPK})$. We depict three types of correctness in turn.

1. Assume a statement $x = (p||m_r, \mathsf{MPK})$, a witness $w = (pk, sk, \sigma)$. Since $\sigma = \mathsf{S.Sign}(sk, \mathsf{MPK})$, we have $\mathsf{S.Verify}(pk, \sigma, \mathsf{MPK}) = 1$ by the correctness of signature scheme S. Since sk exactly corresponds to pk such that $pk = \mathsf{F}(sk)$, the correctness is ensured by the function F. Since $\eta = \mathsf{Z.Prover}(x, w, \mathsf{PP})$, we have $\mathsf{Z.Verifier}(x, \pi, \mathsf{PP}) = 1$ by the correctness of algorithm Z. Thus, the *verification correctness* is ensured.
2. Let $\mathcal{H}$ denote a secure hash function, $p||m_1$, $p||m_2$ be two messages with a common prefix p. Assume π_1, π_2 are the corresponding authentication tokens, we can achieve the *linking correctness.* If $\mathsf{Verify}(p||m_i, \pi_i, \mathsf{MPK}) = 1$ for $i = 1, 2$. Since $t_1 = \mathcal{H}(p, sk)$, there must be the same value of $\mathcal{H}(p, sk)$ in π_1, π_2 such that the two messages are linked by the correctness of the hash function $\mathcal{H}$.
3. Meanwhile, the *tracing correctness* is also similar. If $\mathsf{Verify}(m_i, \pi_i, \mathsf{MPK}) = 1$ for $i = 1, 2$; $\mathsf{Link}(m_1, m_2, \pi_1, \pi_2) = 1$. Since $t_2 = \mathcal{H}(p, pk, sk) + m \cdot sk$, thus, π_1, π_2 have the same value of $\mathcal{H}(p, pk, sk)$ due to the correctness of the hash function $\mathcal{H}$, and the remaining part of t_2 will be used for inferring the public key pk.

Security. We list the security theorems in the following aspects.

Theorem 1 (Unforgeability). *A* PLTAA *scheme is unforgeable under the random oracle model.*

Proof. We require any uncertified attacker cannot forge one's identity to authenticate a message due to the lack of a secret key which corresponds to a public key and a certificate. Assume a user with (pk, sk) obtains a certificate σ. For a message $p||m$, the user calls Auth to produce an authentication token $\pi = (t_1, t_2, \eta)$, which will be successfully verified by invoking Verify. For the authentication token π, it consists two hash values and one proof, which is the only transcript that can be seen by the attacker. The two hash values are computed by $\mathcal{H}(p, sk)$ and $\mathcal{H}(p, pk, sk)$. In order to forge a valid authentication on $p||m$, the attacker needs to obtain the secret key sk that can be extracted from randon oracle queries. Therefore, even an adversary obtains the pk and σ, there is no way to forge a π' using a forged sk' such that it can pass the verification without the corresponding sk.

Theorem 2 (Anonymity). *A* PLTAA *scheme is anonymous under the random oracle model.*

Proof. We require that the attacker cannot tell the user after he got the authentication transcript. Assume a user with (pk, sk) obtains a certificate σ and create an authentication token $\pi = (t_1, t_2, \eta)$ on the message $p||m$. But t_1, t_2 in the authentication transcript can be viewed as random values, which are unable to discover one's sk. That is to say, none can recognize the difference between $\mathcal{H}(p, sk)$, $\mathcal{H}(p, pk, sk) + m \cdot sk$ and a random value.

Theorem 3 (Linkability). *A* PLTAA *scheme is linkable under the random oracle model.*

Proof. The linkability is ensured by generating the tag t_1 in Auth. An attacker with (pk, sk) chooses two message $p||m_1$, $p||m_2$ sharing a common prefix p, invokes Auth to authenticate them and obtains π_1, π_2. Apparently, π_1, π_2 will pass verification by calling Verify. However, $\mathcal{H}(p, sk)$ in tag t_1 of the authenticate transcript makes all messages authenticated by using an identical sk to be linked, i.e., $p||m_1$ and $p||m_2$ will be linked.

Theorem 4 (Accountability). *A* PLTAA *scheme is accountable under the random oracle model.*

Proof. The accountability is achieved by creating the tag t_2 in Auth. An attacker with (pk, sk) chooses two message $p||m_1$, $p||m_2$ sharing a common prefix p, calls Auth to authenticate them and obtains the corresponding authentication tokens π_1, π_2. Apparently, π_1, π_2 will pass verification by calling Verify, and m_1, m_2 will be linked by calling Link. However, π_1, π_2 will get the same value of $\mathcal{H}(p, pk, sk)$ in tag t_2 if using the same (pk, sk). Accordingly, the pk is exposed with the help of the rest part of tag t_2, i.e., $m_i \cdot sk$ for i = 1, 2.

Summarizing the above aspects, we have:

Theorem 5 (Security). *A* PLTAA *scheme is secure provided the discrete logarithm problem is difficult under the random oracle model.*

References

1. Chow, S.S.M., Liu, J.K., Wong, D.S.: Robust receipt-free election system with ballot secrecy and verifiability. In: NDSS, vol. 8, pp. 81–94 (2008)
2. Kiayias, A., Yung, M.: Self-tallying elections and perfect ballot secrecy. In: Naccache, D., Paillier, P. (eds.) PKC 2002. LNCS, vol. 2274, pp. 141–158. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45664-3_10
3. Lee, B., Boyd, C., Dawson, E., Kim, K., Yang, J., Yoo, S.: Providing receipt-freeness in mixnet-based voting protocols. In: Lim, J.-I., Lee, D.-H. (eds.) ICISC 2003. LNCS, vol. 2971, pp. 245–258. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24691-6_19
4. Abe, M.: A secure three-move blind signature scheme for polynomially many signatures. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 136–151. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44987-6_9
5. Hirt, M., Sako, K.: Efficient receipt-free voting based on homomorphic encryption. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 539–556. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-45539-6_38

6. Liu, J.K., Wei, V.K., Wong, D.S.: Linkable spontaneous anonymous group signature for Ad Hoc groups. In: Wang, H., Pieprzyk, J., Varadharajan, V. (eds.) ACISP 2004. LNCS, vol. 3108, pp. 325–335. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-27800-9_28
7. Tsang, P.P., Wei, V.K., Chan, T.K., Au, M.H., Liu, J.K., Wong, D.S.: Separable linkable threshold ring signatures. In: Canteaut, A., Viswanathan, K. (eds.) INDOCRYPT 2004. LNCS, vol. 3348, pp. 384–398. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-30556-9_30
8. Li, M., et al.: Crowdbc: a blockchain-based decentralized framework for crowdsourcing. IEEE Trans. Parallel Distrib. Syst. **30**(6), 1251–1266 (2018)
9. Tsang, P.P., Wei, V.K.: Short linkable ring signatures for E-Voting, E-Cash and attestation. In: Deng, R.H., Bao, F., Pang, H.H., Zhou, J. (eds.) ISPEC 2005. LNCS, vol. 3439, pp. 48–60. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-31979-5_5
10. Wei, V.K.: Tracing-by-linking group signatures. In: Zhou, J., Lopez, J., Deng, R.H., Bao, F. (eds.) ISC 2005. LNCS, vol. 3650, pp. 149–163. Springer, Heidelberg (2005). https://doi.org/10.1007/11556992_11
11. Fujisaki, E., Suzuki, K.: Traceable ring signature. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 181–200. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-71677-8_13
12. Kiayias, A., Tsiounis, Y., Yung, M.: Traceable signatures. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 571–589. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24676-3_34
13. Smart, M., Ritter, E.: Remote electronic voting with revocable anonymity. In: Prakash, A., Sen Gupta, I. (eds.) ICISS 2009. LNCS, vol. 5905, pp. 39–54. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-10772-6_5
14. Ling, L., Liao, J.: Anonymous electronic voting protocol with traceability. In: 2011 International Conference for Internet Technology and Secured Transactions, pp. 59–66. IEEE (2011)
15. Lu, Y., Tang, Q., Wang, G.: Zebralancer: private and anonymous crowdsourcing system atop open blockchain. In: 2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS), pp. 853–865. IEEE (2018)
16. Hinterwälder, Gesine, Zenger, Christian T., Baldimtsi, Foteini, Lysyanskaya, Anna, Paar, Christof, Burleson, Wayne P.: Efficient E-Cash in practice: NFC-based payments for public transportation systems. In: De Cristofaro, Emiliano, Wright, Matthew (eds.) PETS 2013. LNCS, vol. 7981, pp. 40–59. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-39077-7_3
17. Malina, L., Smrz, J., Hajny, J., Vrba, K.: Secure electronic voting based on group signatures. In: 2015 38th International Conference on Telecommunications and Signal Processing (TSP), pp. 6–10. IEEE (2015)
18. Moran, T., Naor, M.: Split-ballot voting: everlasting privacy with distributed trust. ACM Trans. Inf. Syst. Secur. (TISSEC) **13**(2), 16:1–16:43 (2010)
19. Adida, B.: Helios: Web-based open-audit voting. USENIX security symposium **17**, 335–348 (2008)
20. Culnane, C., Schneider, S.: A peered bulletin board for robust use in verifiable voting systems. In: 2014 IEEE 27th Computer Security Foundations Symposium, pp. 169–183. IEEE (2014)
21. Dini, G.: A secure and available electronic voting service for a large-scale distributed system. Future Gener. Comput. Syst. **19**(1), 69–85 (2003)

22. Burton, C., Karunasekera, S., Harwood, A., Stanley, D., Ioannou, I.: A distributed network architecture for robust internet voting systems. In: Wimmer, M.A., Traunmüller, R., Grönlund, Å., Andersen, K.V. (eds.) EGOV 2005. LNCS, vol. 3591, pp. 300–308. Springer, Heidelberg (2005). https://doi.org/10.1007/11545156_29
23. Gibson, J.P., Lallet, E., Raffy, J.-L.: Engineering a distributed e-voting system architecture: meeting critical requirements. In: Giese, H. (ed.) ISARCS 2010. LNCS, vol. 6150, pp. 89–108. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13556-9_6
24. Kyrillidis, L., Cobourne, S., Mayes, K., Dong, S., Markantonakis, K.: Distributed e-voting using the smart card web server. In: 2012 7th International Conference on Risks and Security of Internet and Systems (CRiSIS), pp. 1–8. IEEE (2012)
25. Chondros, N., et al.: D-demos: a distributed, end-to-end verifiable, internet voting system. In: 2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS), pp. 711–720. IEEE (2016)
26. Groth, J.: Efficient maximal privacy in boardroom voting and anonymous broadcast. In: Juels, A. (ed.) FC 2004. LNCS, vol. 3110, pp. 90–104. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-27809-2_10
27. Hao, F., Ryan, P.Y., Zieliński, P.: Anonymous voting by two-round public discussion. IET Inf. Secur. **4**(2), 62–67 (2010)
28. Zhao, Z., Chan, T.-H.H.: How to vote privately using bitcoin. In: Qing, S., Okamoto, E., Kim, K., Liu, D. (eds.) ICICS 2015. LNCS, vol. 9543, pp. 82–96. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-29814-6_8
29. McCorry, P., Shahandashti, S.F., Hao, F.: A smart contract for boardroom voting with maximum voter privacy. In: Kiayias, A. (ed.) FC 2017. LNCS, vol. 10322, pp. 357–375. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70972-7_20
30. Yang, X., Yi, X., Nepal, S., Han, F.: Decentralized voting: a self-tallying voting system using a smart contract on the ethereum blockchain. In: Hacid, H., Cellary, W., Wang, H., Paik, H.-Y., Zhou, R. (eds.) WISE 2018. LNCS, vol. 11233, pp. 18–35. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-02922-7_2
31. Yu, B., et al.: Platform-independent secure blockchain-based voting system. In: Chen, L., Manulis, M., Schneider, S. (eds.) ISC 2018. LNCS, vol. 11060, pp. 369–386. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-99136-8_20
32. Zhang, W., et al.: A privacy-preserving voting protocol on blockchain. In: 2018 IEEE 11th International Conference on Cloud Computing (CLOUD), pp. 401–408. IEEE (2018)
33. Sampigethaya, K., Poovendran, R.: A framework and taxonomy for comparison of electronic voting schemes. Comput. Secur. **25**(2), 137–153 (2006)
34. Cooke, R., Anane, R.: A service-oriented architecture for robust e-voting. SOCA **6**(3), 249–266 (2012)
35. Li, H., Kankanala, A.R., Zou, X.: A taxonomy and comparison of remote voting schemes. In: 2014 23rd International Conference on Computer Communication and Networks (ICCCN), pp. 1–8. IEEE (2014)
36. Chaum, D.: The dining cryptographers problem: unconditional sender and recipient untraceability. J. Cryptology **1**(1), 65–75 (1988)
37. Pan, H., Hou, E.S.H., Ansari, N.: An e-voting system that ensures voter confidentiality and voting accuracy. In: Proceedings of IEEE International Conference on Communications, pp. 825–829. IEEE (2012)

38. Zhang, F., Kim, K.: ID-based blind signature and ring signature from pairings. In: Zheng, Y. (ed.) ASIACRYPT 2002. LNCS, vol. 2501, pp. 533–547. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-36178-2_33
39. Ben-Sasson, E., Chiesa, A., Genkin, D., Tromer, E., Virza, M.: SNARKs for C: verifying program executions succinctly and in zero knowledge. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8043, pp. 90–108. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40084-1_6