Subsenate

Doron Zarchy

Abstract

E-voting systems, which operate over a decentralized platform, may not scale well when used for too many elections or when unprecedented numbers of users participate in the process. As an alternative, we propose Subsenate which is a zero knowledge-based scheme for voting over blockchain which grants permission to vote only to a random subset of the registered voters. Subsenate is robust in its ability to defend against election outcome manipulation. Subsenate keeps the identities of the users who are selected as eligible voters hidden both before and after the vote. Subsenate incentivises truthful behavior by rewarding only the voters who vote with the majority.

1 Motivation

A decentralized and secure E-voting system is a technology with a great amount of potencial. The outcome in prediction markets, open disputes, and functionalities of smart contract that involve external events could all be determined by election.

However, a system that processes many elections at a high speed and involves many users may not scale well as doing so may require interaction amongst too many users and may consume too much bandwidth. To address this, we propose a scheme called Subsenate for choosing a community of judges to resolve the challenge of scalability. Subsenate only counts the votes of a random subset of the users which has been agreed to in advance.

The chance to become elected is equal among all the registered users (note that each "user" is associated with some unique public key). The system is secure against malicious adversaries who attempt to increase the chance of being selected as eligible voters. The system incentivises truthful votes by rewarding only the users who vote with the majority . The selected voters should not be exposed to any other user (including the selected voters) to prevent bribery or coercion.

The requirements for the system are the following:

1. a decentralized mechanism for selecting a random set of the users where each user has equal chance to become elected
2. hiding the identities of the eligible voters
3. hiding the identities of the voters (the eligible voter who practiced their right to vote)

4. rewarding honest voters
5. hiding the identity of election creator

2 Overview of the protocol

We now describe a very high level overview of the protocol:

1. election creator sends a transaction to "creation election" smart contract that contains the list of candidates, majority type (simple or super) , a deposit, and a URL to website that describe the background of the election. The deposit is used for rewarding the honest voters
2. registration to the election is done via sending a special transaction to a smart contract
3. after an election is set, the voters compute a random seed
4. subset of eligible voter is selected according ot the random seed
5. each selected user publish a proof attesting membership of the subset of eligible voters
6. miners approve all the votes whose proofs are verified
7. the election outcome is set to be the candidate that gets the majority of the ballots of the approved voters
8. miners unlock the collateral of the election creator and send the coins to the "winners" of the election

3 Assumptions

Although Subsenate grants equal chance to all eligible voters, we assume that the voting power splits proportionally to the stakes. We do it by assuming that anyone who register the system utilize it's entire stakes for voting (by assuming the existence of wallets which split the coins to different addresses keys) –TBD, privately distribute the deposit according to the stakes, i.e., the reward split according to the stakes. The security of the system is based on the assumption that the majority of the stakes is honest. In addition we assume that the originators of the proofs are unknown. Since some blockchains (such as Ethereum) requires signing the transactions, we assume a mixed network that hides the massage senders – elaborate–.

4 Election Creation and Registration

An election creator sends a transaction to a smart contract which specifies the following:

1. a deposit of the election creator (which will be later distributed among the honest voters)
2. the type of election rule can be one of the following:
   (a) simple majority, i.e., candidate who received most of the ballots (if there are two candidate then its a simple majority)
   (b) super-majority, defined by a parameter α. The election is finalized only if more than α-fraction of the voters voted to some candidate.
3. the duration where voters are allowed to vote. Specifically, the smart contract sets an interval $[T_1, T_2]$ for which voter's vote is considered only between blocks t_1 and t_2.
4. the size, in expectation, of the subset of voters, denoted R
5. (*) a URL whose website contains content of the election/dispute.
6. list of candidates V.

Registering to the election requires sending a transaction to a smart contract that contains a commitment to the voter's identity and a solution to some cryptographic puzzle (e.g., finding a preimage of some hash function) which serves as an anti-spam.

5 Voting

The voter cannot send the vote in plaintext as it would expose its identity, hence a commitments of both to the voter's identity as well as to the vote are published (Subsenate does not expose the link between the voter and his commitment). The attestation to the commitment is done via zkSNARK.

The voting proceeds in two phases: commitment to voter's identity and commitment to the vote. Let $\mathbb{F}_p$ be a field of prime order p. The user compute the commitment cm to voter's identity as follows:

1. select a secret key sk uniformly from $\mathbb{F}_p$
2. select nonces t_1 and t_2 uniformly from $\mathbb{F}_p$
3. computes the commitment $cm = H(sk, t_1, t_2, u)$
4. sends cm and u to the smart contract

The actual voting is done by publishing a vote commitment c_v that is computed as $c_v = H(sk, v, SR, u, pk_{new})$ where

1. $v \in V$ is the vote.
2. pk_{new} is a derivation from the new public key (for the subsequent address), i.e., $pk_{new} = g^{sk}$, where $g \in \mathbb{G}$ is a generator of a multiplicative group
3. select u uniformly from $\mathbb{F}_p$. u is a nullifier which guarantees that all votes are unique (similar to the use in Zcash)
4. SR is a serial number of the election

5.1 Selecting the Subset

A user is considered as eligible voter if $H(S, t_1) < R$ where t_1 is the nonce that the user committed to in phase one, and S is a random seed. S should be secure against a manipulation of an adversary. To guarantee this security it is computed by multiple parties as follows:

1. k users publish their commitment $c_1, ..., c_k$ (which are hashes of random values) at time interval $[T_1, T_2]$ (where T_1 and T_2 are determined by the smart contract of the election creator)
2. users publish $o_1, ..., o_l$ which are the preimages, WLOG, of $c_1, ..., c_l$, at time intervals $[T_3, T_4]$
3. each miner computes $S = H(o_1, ..., o_l)$, where H is some collision resistant hash function.

5.2 zkSNARK statement

The voter generates a zkSNARK proof $\Pi = (\Pi_1, \Pi_2, \Pi_3, \Pi_4)$ for the following four NP statements:

1. $cm = H(sk, t_1, t_2)$ where cm is the commitment published in the smart contract when the voter registers the election. The voter proves her identity by showing that she knows random nonces t_1 and t_2.
2. $c_v = H(sk, v, SR, u, pk_{new})$, the commitment to the vote v also includes the secret key of the voter, the serial number of the vote, the nullifier, and the public address for which assets from deposit should be transferred.
3. $Root = \mathbb{H}(mp, cm)$. Proving that cm is a member in the set of commitments by proving that the Merkle root is a result of the hashing cm and the witness (Merkle path) . The Merkle root is public.
4. $H(S, t_1) < R$ proves that the voter is eligible.

where the public input is:

1. Root: Merkle root of the commitment set
2. u: a nullifier

3. v: vote
4. pk_{new}: voter's new address
5. N: expected number of voters
6. S: random seed
7. SR: serial number of the vote

the private input:

1. cm: commitment to the vote
2. mp: a Merkle path from cm to root
3. t_1 and t_2: random nonce
4. sk: private key (but not used for signature)

The sender can simply send the proof Π along with nullifier u, vote v, and the serial number of the election. Note that in Ethereum like systems where the sender's address is part of the transaction, the transaction can be used to break anonymity. Therefore a mixed network is required (a simpler solution use a coordinator which acts as an address that obfuscate the voters's addresses, where the coordinator establishes an offchain connection with the voter which allows the voter to send the proof privately to the coordinator who then publishes the proof).

6 Verifying the proof

Each miner that verifies the proof

- constructs a Merkle tree based on the vote commitments $cm_1, ..., cm_k$ listed in the smart contract of the election (with their defined order). and derive Root.
- computes a random seed S
- verify the zkSNARK proof based on the verification key, proof Π and the public parameters.

Then the miners reward the new accounts pk_{new} of the winner voters, i.e., voters whose ballots are within the majority then it receives equal share of the deposit of the election creator.

7 Security (very high level)

The privacy of the system is based on the security of Groth16 []. Users cannot vote more than once as it requires issuing different nullifiers for the same commitment which is as hard as finding collision of CRHF. The rewards received by the honest voters do not break anonymity as all winner users gets equal amount of coins. UC security can be proven via...TBD

We claim that the distribution of the honest voters in the chosen subset does not have a large bias from the distribution of the honest voters among the entire set of the users. Let N be the number of users and r be the size of the subset that is selected for voting. The probability that a majority of the selected users are honest is

$$\sum_{i=\lfloor \frac{r}{2} \rfloor + 1}^{r} \frac{\binom{\frac{N}{2}}{i}\binom{\frac{N}{2}}{r-i}}{\binom{N}{r}}$$

where we assume that at least half of the voters are honest. We now show that this probability is larger than 99.9% for any $N >$ ... and $R >$ ....

Another way to mitigate the bias and disincentivize the adversary attacking the system, e.g., by bribing other users to vote for some (false) candidate and later sell all the stakes, can be done by allowing any the stake transfers to be performed only after sufficient amount of time (through consensus). Assuming that a after a successful attack of the system the value of the coins will decrease to such extent that selling the stakes will be unprofitable to the attacker.