e-Voting Machine

Investor Presentation

S4Y Solutions

Author: Sergey Dolin
sergey@s4y.solutions

2023/2024

The Challenges of the Voting Process

Voting presents two critical challenges:

- **Balancing Transparency and Privacy:** The need to verify each cast vote while maintaining the anonymity of the voter.
- **Voter Identification:** Ensuring that each voter is uniquely and securely identified.

Transparency vs Privacy

Unsolvable contradiction

The challenge of verifying a vote while maintaining complete privacy seems like an unsolvable contradiction

Solution

Allowing voters to cast multiple votes lets them verify their choice while keeping the actual vote undiscoverable

Raises another problem

Allowing multiple votes opens the door for potential fraud

Revenue: Secure, Transparent Cast Ledger Service

Voter Identification

Conventional Voting

Paper ballots does not solve the problem of the identification completely

e-Voting

Neither e-voting can identify a voter with absolute certainty, due to the limitations in accessing personal data, but it is possible to provide a predictable level of certainty with different built-in and external services

Revenue: Paid identification services
(emails, biometric, heuristic, government and NGO-issued IDs, etc)

e-Voting Specific Vulnerability

Conventional Voting

The voting booth provides a level of protection against external influence on how the voter casts their vote.

e-Voting

In electronic voting, it is impossible to protect the voter from being forced to vote in a specific way.

Multiple Voting

Multiple voting allows the voter to cast their vote freely, without external pressure

Extra benefits

Supplementary and Related Services:

- **Real-time Vote Tally:** Exit-polling Service
- **Historical data:** API to access historical voting data
- **Extra security:** Provide services with private access

Implementation details: Ledgers

Altering votes issue

Allowing vote alterations implies modifications to records, which can undermine trust in the service.

Immutable ledger

This is addressed by an add-only ledger and a two-step vote change: negating the old vote and casting a new one.

Raises another problem

Consequence votes introduce the need to link back to the previous vote, which compromises voter anonymity as it can be traced.

Implementation details: Smart contracts

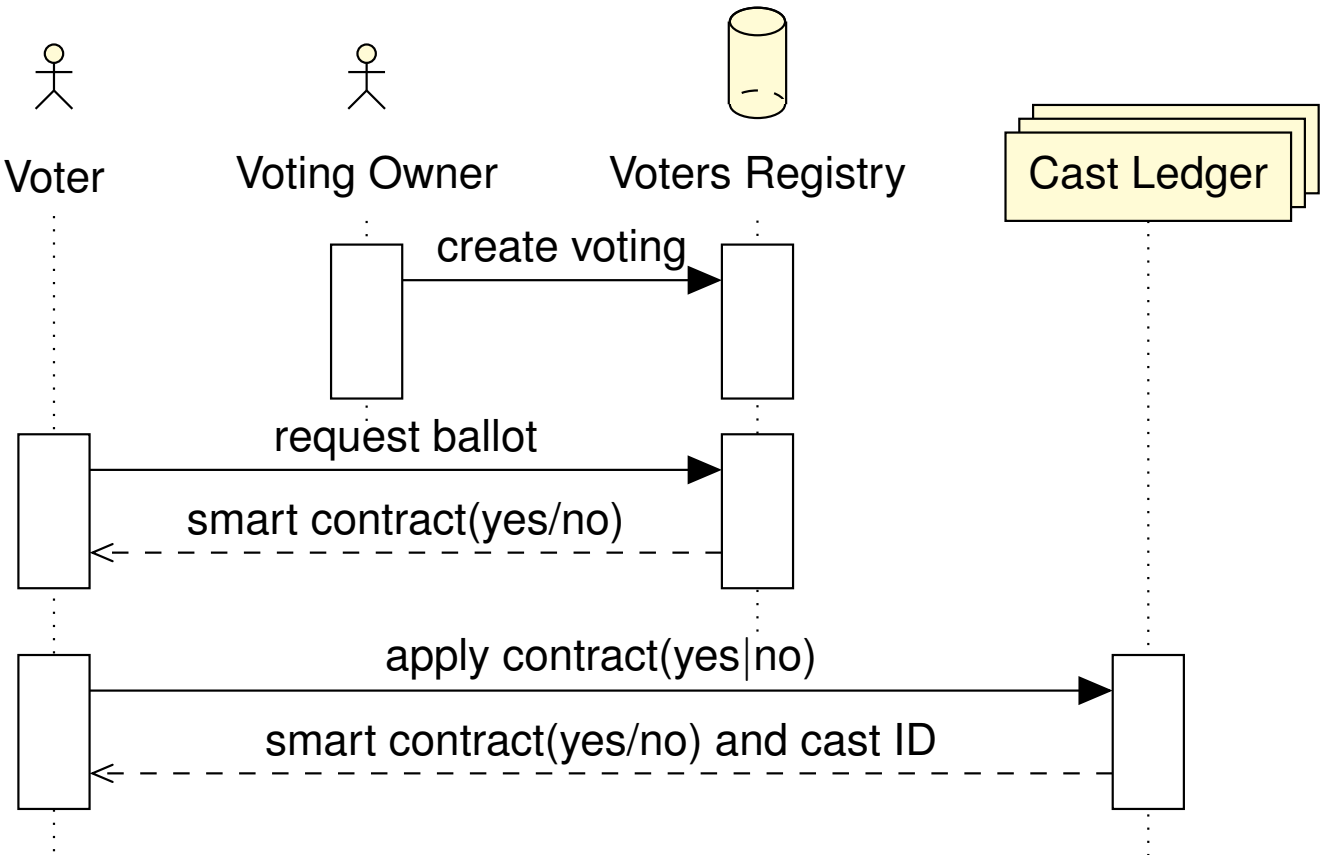
Don’t keep the link to the previous vote

It is not necessary to link to the previous vote in order to negate it. Regardless of when or who cast the vote, simply knowing which vote it was and adding its negation to the ledger will effectively remove the previous vote.

Voting with deffered self-contained code snippet

Smart contracts are chosen for this purpose—these are secured, signed code snippets executed in an isolated environment. This approach was selected due it fits well with existing infrastructure and industry standards, making it both practical and reliable.

High-level overview of the voting flow



Ensuring trust in voting

Transparency is the key to ensuring trust in the voting process.

- **Voters Registry:** Shows only the issuance of ballots without linking votes to individual voters
- **Cast Ledger and Issued Contracts Ledgers:** Make it possible to verify only system issued ballots were used for voting
- **Open Smart Contract Code:** Since no code other than the smart contract can affect the voting process, it ensures trustworthiness in e-voting
- **3rd Party Provided Ledgers:** Hosting the ledgers and running smart contracts on public blockchains eliminates doubts about data tampering

Enhancing Security

Moderate level of security assumes that each cast (or applying the smart contract) has its own unique ID, allowing voters to verify that their vote is recorded in the ledger.
Each vote has a unique ID, so the voter can confirm his vote was counted. While this ID doesn't reveal which specific vote he made (since he can vote multiple times), seeing that a new vote was cast suggests he might have changed his mind.
It's possible to offer a service with anonymous contracts. However, this means no one can verify a specific vote, so both the smart contracts and ledgers must be completely trustworthy.

Implementation details: Infrastucture

The terms “Smart Contracts” and “Ledgers” might suggest the system relies on blockchain technology. While blockchain is a suitable option, it’s not the only one. The system can also be implemented with a traditional, audited database or a secure cloud-hosted server. Even if blockchain is used, it doesn’t have to be public. A self-hosted, private blockchain could be considered in the actual implementation.

Implementation details: Voters identifications

Although this presentation focuses on the voting process, the initial stage of issuing ballots is crucial.
It is not covered here because it will be implemented as a separate service. This service will support plugins and be extendable based on client needs and available data sources.

Implementation details: Microservices

The system should be built from loosely coupled microservices connected by a common protocol, allowing components to be swapped based on client needs.

Current status

At the time of writing, the project includes a Proof of Concept Lua-based reference implementation available in a public repository on GitHub:
`https://github.com/s4ysolutions/e-voting-machine`.