Removing Trusted Tallying Authorities

Self-Enforcing E-Voting over Ethereum

Authors: *Patrick McCorry, Ehsan Toreini, Maryam Mehrnezhad*

Table of Contents

"I consider it completely unimportant who in the party will vote, or how; but what is extraordinarily important is this — who will count the votes, and how."

– Reported from Joseph Stalin [1]

1. Introduction

Paper-based voting

In paper-based voting, tallying is a critical process where the winner of an election is determined. When a voter inserts the completed ballot into the box, they lose sight of the ballot and have to trust election authorities to faithfully record and tally ballots. But corrupted authorities may modify, miscount or exclude the voter's ballot without the voter's knowledge. The lack of assurance on the tallying integrity is one major cause for disputes in the aftermath of an election.

Modern e-voting products

In the modern digital era, e-voting products are being adopted by many countries to allow voters to cast ballots on a touch-screen direct-recording electronic (DRE) machine or over the Internet. Similar as before, voters have to trust election authorities to faithfully record and tally their electronic ballots. However, as compared with tampering with physical ballots, it is much easier for a single corrupted authority to tamper with the electronic records and tally.

Academic research

The state-of-the-art in the field of e-voting research concerns voting systems that are end-to-end (E2E) verifiable [2]. Being E2E verifiable means that voters are able to verify if their votes are cast as intended, recorded as cast and tallied as recorded. As the verification covers from the start of casting a vote to the end of receiving the tally, this gives the name "End-to-End verifiable".

To mimic the role of trusted counting staff in paper-based voting, almost all of the E2E voting systems assume tallying authorities (TAs), who are trustworthy individuals with computing and cryptographic expertise tasked to perform the tallying operation. However, voters must trust the TAs do not collude all together, as then they can learn each individual vote. The fact that TAs have such power presents a deterring effect on some voters when choosing their favoured candidates.

Our vision

In our vision, we believe a future-generation e-voting system should be one that provides E2E verifiability *without* depending on any privileged group of people who act as tallying authorities. In other words, the system should be "**self-enforcing**". This is highlighted in Figure 1.

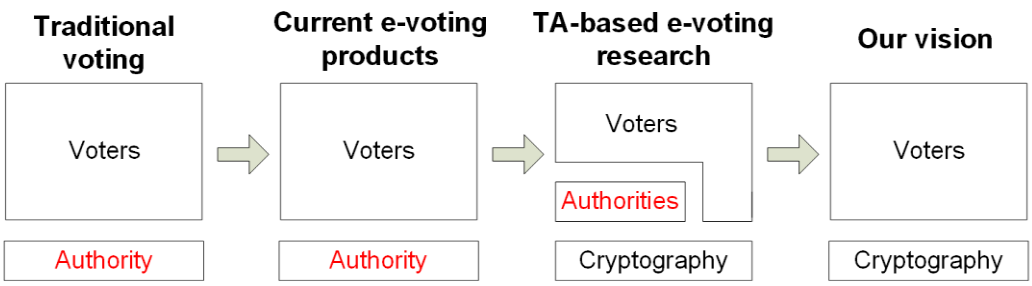

Figure 1: Evolution of the Trust on Tallying Authorities

Our confidence in the feasibility of this vision builds on several existing "self-enforcing" e-voting protocols, namely Open Vote Network (**OV-net**) [3], Direct Recording Electronic with integrity (**DRE-i**) [5] and DRE-i with enhanced privacy (**DRE-ip**) [4]. OV-net is designed for small-scale boardroom voting, while DRE-i and DRE-ip are for national scale elections.

These protocols—in fact all verifiable e-voting protocols—require a public bulletin board where cryptographic data is published for public verification. The publication of data on the bulletin board must be append-only. If the previous audit data can be retrospectively modified, the assurance on the tallying integrity will be lost.

A practical public bulletin board

In this challenge, we investigate Bitcoin [6] and its underlying public ledger, the blockchain, to identify if it can be used as a public bulletin board for electronic voting. Bitcoin's blockchain is immutable and censorship resistant which are desirable properties for an e-voting public bulletin board. Unfortunately, it is only a global singleton database that can store data, and is limited in its support for programming capability, which is needed to enforce the execution of the voting protocol.

Among several existing blockchain systems, we choose Ethereum's blockchain [7] for the proof-of-concept implementation of our e-voting solution. Conceptually, Ethereum is a global singleton computer that can store and execute programs ('smart contracts'). The execution transcripts of these contracts are stored in the blockchain and verified by Ethereum's underlying peer-to-peer (P2P) network. This decentralised P2P network enforces the correct execution of the programs without involving trusted third parties, hence the Ethereum blockchain is also considered "self-enforcing".

In this report, we demonstrate a proof-of-concept implementation of OV-net [3], an efficient self-enforcing e-voting protocol, over Ethereum for the first time.

2. Our Solution: Open Vote Network

Open Vote Network is a decentralized two-round voting scheme [3]. For a single candidate election with the Yes/No choice, this protocol can be described as follows (for the multiple candidate version see [3]). First, all n voters agree on (G, g) where G is a cyclic group of prime order q, and g is a generator in G. Each voter P_i chooses a secret value x_i uniformly at random from $[0, q-1]$.

Round 1: every voter P_i publishes g^{x_i} and a Schnorr Zero Knowledge Proof (ZKP) for proving the knowledge of x_i. At the end of this round, every voter validates all ZKPs, and computes:

$$g^{y_i} = \prod_{j<i} g^{x_j} / \prod_{j>i} g^{x_j}.$$

Round 2: every voter P_i publishes $g^{x_i y_i} g^{v_i}$ and a one-out-of-two ZKP for proving that v_i is either 0 or 1 (for No and Yes respectively). At the end of this round, anyone who observes the protocol can tally the number of ones by computing:

$$\prod_i g^{x_i y_i} g^{v_i} = g^{\sum_i x_i y_i} g^{\sum_i v_i} = g^{\sum_i v_i}.$$

The above protocol works based on the cancellation of random factors at the tallying phase i.e., $\sum_i x_i y_i = 0$ where by the definition $y_i = \sum_{j<i} x_j - \sum_{j>i} x_j$ (see Round 1). As an example, assume n = 4, the random factors will be cancelled as shown in Figure 2. From $g^{\sum_i v_i}$, anyone can compute the tally $\sum_i v$ by exhaustive search.

Example

Assume $n = 4$.

$$\begin{aligned} \sum_i x_i y_i = \quad & \qquad\qquad\quad - x_1x_2 - x_1x_3 - x_1x_4 \\ & + x_2x_1 \qquad\quad - x_2x_3 - x_2x_4 \\ & + x_3x_1 + x_3x_2 \qquad\quad - x_3x_4 \\ & + x_4x_1 + x_4x_2 + x_4x_3 \qquad\qquad = 0. \end{aligned}$$

Figure 2: An example of random factor cancellation

This scheme assumes an authenticated public channel available for every voter. Using a public bulletin board is commonly suggested to realize such a channel. However, in practice, implementing such a secure bulletin board has remained a technical challenge. We believe the blockchain holds the key to this problem and will demonstrate the feasibility by presenting a concrete proof-of-concept implementation.

3. The Proof-Of-Concept Implementation

In our implementation, voters need to connect to Ethereum's underlying peer-to-peer (P2P) network as shown in Figure 3, and their identities are represented by Ethereum accounts. These accounts are simply public-private key pairs that have been locally generated on the voters' machines, and should have a positive balance of ether (Ethereum's currency). The voter can compute a digital signature using their Ethereum account to prove the authenticity of data that they send during the voting process.

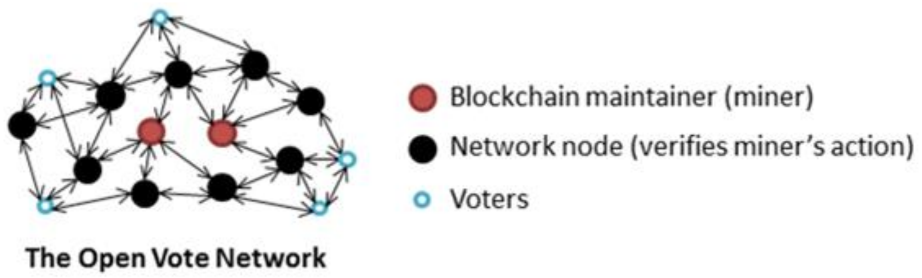

Figure 3: How voters connect to Ethereum's underlying peer to peer network

Implementation

Our proof-of-concept implementation is written in Ethereum's solidity language [8]. We have implemented the system's user interface using HTML5 and JavaScript where it has three different views: voter page, Election Authority page, and a live feed page (see Figures 7-10 in Appendix for screenshot examples). These interfaces are designed to ease the interaction between the voters, Election Authority and the Ethereum network while enabling them to observe the voting procedure.

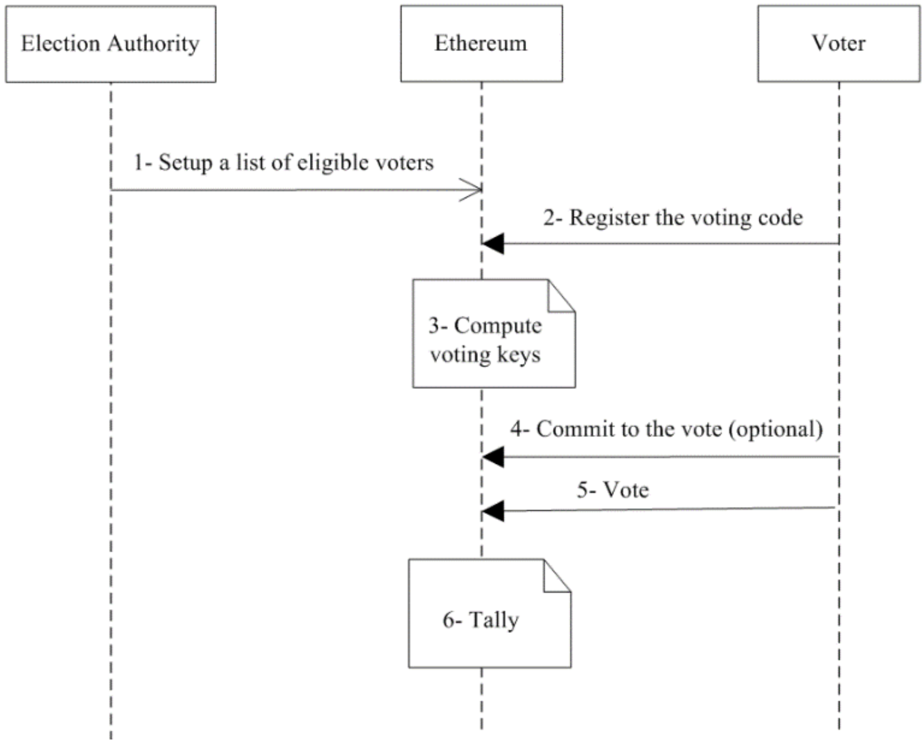

Figure 4: The Sequence diagram of our implementation

The steps of the system are explained in Figure 4.

- **SETUP.** The Election Authority establishes the list of eligible voters and informs Ethereum to transit to the signup phase. In addition to the election question, and in order to assure the voters that the election will happen in a timely manner, a list of start and end times for each phase is sent to the network too.

- **Register.** The voter participates in the first round of OV-net by registering their ballots for the election. We have implemented Schnorr ZKP based on [9]. The registration ballot is accepted by the network once the ZKP is verified successfully. When the registration deadline is past, the Election Authority informs Ethereum to finish the registration phase.

- **COMPUTE.** Ethereum computes each voter's voting key (i.e., g^{y_i}). Each voter can retrieve their voting key from Ethereum before casting their vote.

- **COMMIT (Optional).** The voter can send a 'commitment' of their encrypted vote to the Ethereum network. The commitment is a one-way hash of the round-2 message. Without this phase, in the second round of the protocol the final voter may privately compute the tally before sending their vote and this might influence the candidate they choose. Sending a 'commitment' is the equivalent to posting the vote in a sealed envelope to the network, and only revealing the votes once all sealed envelopes have arrived.

- **VOTE.** The voter participates in the second round of OV-net by sending an encrypted vote and a one-out-of-two ZKP which we have implemented based on [19]. The vote is accepted into the blockchain only if the ZKP is verified successfully.

- **TALLY.** The tally is computed by the Ethereum network using the tally computation method defined in OV-net.

Technicalities of the Ethereum platform

During the implementation, we encountered several technical difficulties.

First, Ethereum only supports 256-bit unsigned integers. For this reason, we chose to implement the protocol over an elliptic curve instead of a finite field. Unfortunately, Elliptic Curve cryptography is not natively supported yet and this required us to find an external library to perform the computation. This library must be stored in the blockchain alongside our program, which led to our initial voting contract being too large to store on the network. To resolve this issue, we had to separate our program into two smart contracts: one 'voting contract' for computing votes and verifying ZKPs and the other 'cryptography contract' for creating ZKPs (see Figure 12 in the Appendix). Note that any computation performed on Ethereum requires 'gas' which can be purchased using 'ether'. Each block has a gas limit that corresponds to the maximum amount of computation allowed.

Second, the call stack of a program has a hard-coded limit of 1024 stack frames [10]. This limits the amount of local memory available, and the number of function calls allowed. These limitations led to difficulty while implementing the 1-out-of-2 ZKP as the temporary variables typically required exceeded the hard-coded limit. We had to use variables extremely sparingly to make the program work.

Third, there exist few debugging tools for these smart contracts. The best practice is to create an ‘Event’ that logs data along with the contract. These events need to be incorporated into the program before compiling the contract, and they do not allow running the code step by step.

Finally, the random numbers used for the ballot and casting the vote need to be stored on the voter's local machine. This is important to ensure that if the voter’s web browser crashes or is accidentally closed, then the random number is not lost. To this end, we built a standalone Java program that generates the random numbers on the voter’s local machine, and the voter is requested to upload those numbers as ‘voting code’ into the voting page.

Cost analysis

Our prototype of OV-net was tested using Ethereum’s official test network [7] with 40 voters to assess the cost of running an election. The voting and cryptography contracts cost £0.78 and £0.50 respectively to store on the blockchain, and 125 Ethereum transactions (see Figure 11 in the Appendix) to run the election with a total cost of £27.72. As shown in Figure 5, the average cost is £0.69 per voter, which is lower than the typical cost of running a paper-based election (see Appendix: Table 3 for a detailed breakdown of cost and Table 4 for comparisons with the reported costs in real-world elections). The most expensive operations include the voter registering their ballot (£0.14, i.e., 15% of a block’s available gas) and casting their vote (£0.49, i.e., 53% of a block’s available gas). This suggests that within one block (generated approximately every 12 seconds) only six voters can register for the election, and only one vote can be cast per block using the current Ethereum network.

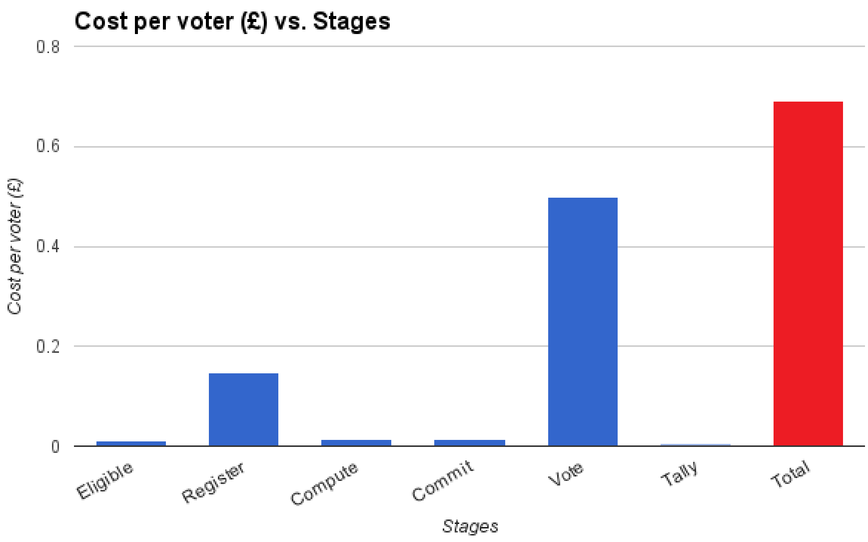

Figure 5: The cost of voting using our system

4. Scaling Up to National Elections

Scaling up our solution to national elections requires addressing limitations in both Ethereum's blockchain and the e-voting protocol.

First, using Ethereum as deployed today, only one vote can be cast in one block. Given that each block is generated every 12 seconds, this means only five votes per minute can be cast over the blockchain. Take the 2011 UK Referendum an example. For 5.2 million votes (the number of postal votes in that election), it would require 722 days for all votes to be recorded into Ethereum's blockchain.

To support national scale elections, a dedicated Ethereum-like blockchain will be required. Such a blockchain will provide a consistent global database that all voters have access to and guarantee that all inserted data remain immutable. Election audit data sent to the blockchain will be verified by independent validators who act as 'miners' and get awarded for verifying the audit data and maintaining the blockchain. Each block should allow storing more votes by increasing the gas limit and new blocks may be generated at a faster speed than the current 12 seconds per block.

Second, another limitation concerns the e-voting protocol. OV-net is decentralized and is designed only for small-scale boardroom voting [3]. To support national-scale elections, we propose using DRE-i [5] and DRE-ip [4], which follow a similar "self-enforcing" idea as the Open Vote Network but use a centralised voting facility (either a web server or a DRE machine) to directly record votes from the voter (without knowing the voter's real identity, which can be ensured through physical or procedural means [4,5]). For both DRE-i and DRE-ip the centralised facilities need to connect to the Ethereum network, as shown in Figure 6.

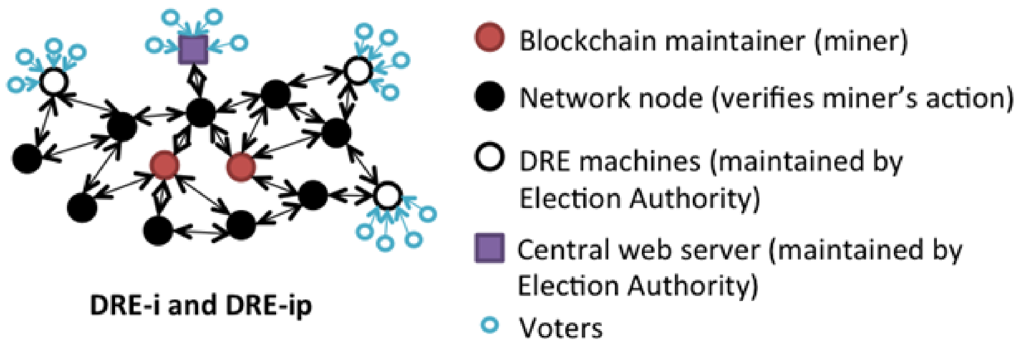

Figure 6: How DRE-i and DRE-ip connect to the Ethereum network

Due to the space limit, we only briefly describe DRE-i [5] and DRE-ip [4]. Both protocols are E2E verifiable voting protocols designed for supporting large-scale elections without tallying authorities. The difference between the two is that DRE-i pre-computes the encrypted ballots before the election while DRE-ip computes the encrypted ballots in real time during voting. The pre-computation has the advantage of minimizing the latency in voting, which makes **DRE-i a suitable choice for Internet voting** since the server must be able to handle many simultaneous vote submissions. By contract, DRE-ip does not perform pre-computation and hence removes the need to securely store the pre-computed ballots. The protocol provides strong guarantee on the vote privacy in the sense that when the DRE machine is completely compromised, the information leakage is minimal as only the partial tally is revealed. These properties make **DRE-ip a suitable choice for polling station voting**.

5. Meeting the Challenge's Criteria

As summarized in Table 1, OV-net satisfies four out of the five criteria set by Kaspersky. The only exception is that it does not prevent voting under duress. This is because voting happens in an **unsupervised environment** and the voter is not guaranteed a private moment to cast their vote. This can be addressed by implementing e-voting under a **supervised environment** at polling stations using DRE-ip; a private moment of voting is assured by the use of a private voting booth. OV-net provides the maximum protection on voter privacy as only a full-collusion that involves all other voters can reveal the vote [3]. The tallying process guarantees that all votes stored on Ethereum's blockchain are included in the final tally, which everyone can compute. The tally is only computable when the final vote has been cast, which effectively hides interim results. Finally, the protocol allows easily adding an 'abstain' option as an additional candidate choice for undecided voters.

Table 1: Open Vote Network vs. challenge's criteria

Criteria	The Open Vote Network
Voter privacy and the ability to count votes	All voters must collude to reveal an individual vote, and the system is self-tallying without needing any trusted tallying authorities.
Problem of voting under duress	Voter has no private moment, and coercion is possible.
Availability of interim results	No interim results available; tally computable only when the last vote is cast.
Undecided Voters	Voter has the option to register for election; empty votes cannot be casted; cast votes cannot be modified and voters can select ‘abstain’.
The voting aftermath	Dispute-free; all election data is publicly verifiable.

In our analysis, we identify three different voting settings: **decentralized Internet voting**, **centralized Internet voting** and **centralized polling station voting**. Accordingly, we present a solution for each of these settings. Our three solutions cover all election scenarios that we know of today. All our solutions allow voters to verify the tallying integrity without having to trust TAs, while the blockchain self-enforces the execution of the voting protocol. As compared with existing voting methods in real-world elections, our solutions provide compelling benefits in terms of voter verifiability and assurance on the tallying integrity, as summarized in Table 2.

Table 2: Summary of comparison on verifiability and tallying integrity

	Decentralized remote voting	Centralized remote voting		Centralized polling station voting			
Schemes	**Open Vote network**	**DRE-i**	Postal	**DRE-ip**	Paper	DRE	DRE with paper audit trail
Voter can verify if vote is cast as intended	✔	✔	✔	✔	✔	✘	✔
Voter can verify if the cast vote is recorded	✔	✔	✘	✔	✘	✘	✘
Voter can verify if votes are tallied as recorded	✔	✔	✘	✔	✘	✘	✘
Assurance on tallying integrity when TAs are all corrupted	✔	✔	✘	✔	✘	✘	✘
Suitable election	**Small-scale**	**Large-scale**	Large-scale	**Large-scale**	Large-scale	Large-scale	Large-scale

6. Conclusion

The Economist and Kaspersky challenged us to build secure digital voting using the blockchain. We found motivation from the realm of cryptocurrencies that has so far successfully removed the need to trust a central bank or institution to maintain a financial ledger. In this challenge, we proposed to remove trusted tallying authority from the election process. To accomplish this goal, we built a prototype of the Open Vote Network protocol and demonstrate that it is a practical solution that works on Ethereum today. The role of

Ethereum is not limited to a simple public bulletin board, but also to enforce the correct execution of the voting protocol.

Two further protocols DRE-i and DRE-ip are described to demonstrate that our approach can scale up to a national election. Most importantly, all solutions are fully verifiable and provide a strong guarantee on the integrity of the tallying results – and by doing so, preserving the integrity of democracy.

Bibliography

[1] Boris Bazhanov, *Memoirs of Stalin's Former Secretary*, Moscow: III Tysiacheletie, 2002.

[2] Syed Taha Ali, Judy Murray, "An Overview of End-to-End Verifiable Voting Systems", Chapter 8 of *Real-World Electronic Voting: Design, Analysis and Deployment*, CRC Press, 2016.

[3] Feng Hao, Peter Ryan, Piotr Zielinski, "Anonymous Voting by 2-Round Public Discussion", IET Information Security, Vol. 4, No. 2, pp. 62-67, 2010.

[4] Siamak F. Shahandashti and Feng Hao, "DRE-ip: A Verifiable E-Voting Scheme without Tallying Authorities", the 21st European Symposium on Research in Computer Security (ESORICS), 2016.

[5] Feng Hao, Matthew Kreeger, Brian Randell, Dylan Clarke, Siamak Shahandashti, Peter Lee, "Every Vote Counts: Ensuring Integrity in Large-Scale Electronic Voting", *USENIX Journal of Election Technology and Systems* (JETS), Vol. 2, No. 3, 2014.

[6] Bitcoin, A payment network and cryptocurrency, https:// bitcoin.org.

[7] Ethereum, a decentralized platform that runs smart contracts, https://www.ethereum.org/.

[8] Solidity, the Ethereum Virtual Machine language, http://solidity.readthedocs.io/en/develop/

[9] Claude P. Schnorr, "Efficient signature generation by smart cards", *Journal of Cryptology*, Vo. 4, No. 3, pp. 161–174, 1991.

[10] Error while compiling: Stack too deep, http://ethereum.stackexchange.com/questions/6061/error-while-compiling-stack-too-deep

[11] Costs of the May 2011 referendum on the UK Parliamentary voting system, http://www.electoralcommission.org.uk/__data/assets/pdf_file/0009/153000/Costs-of-UK-May-2011-UKPVS-referendum.pdf

[12] CACEO Election Costs Study, http://results.caceoelectioncosts.org/

[13] Election cost statistics, Accountability in Colorado Elections, https://www.sos.state.co.us/pubs/elections/ACE/index.html

[14] Summary of North Dakota Election Statistics 1980 - Present, https://vip.sos.nd.gov/pdfs/Portals/statistics-turnout.pdf

[15] 2014 Spring Election: GAB-190NF Election Statistics and GAB-191 Election Specific Cost Reports, http://www.gab.wi.gov/publications/statistics/gab-190/April-2014

[16] Election Expenditure by Central Government (Towards States/Uts Having Legislature) For General Elections 1951-52 – 2009, http://pib.nic.in/elections2014/Tree.aspx

[17] Voting Methods And Equipment by State, https://ballotpedia.org/Voting_equipment_by_state

[18] Election Commission of India, http://eci.nic.in/eci_main1/evm1.aspx

[19] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers, "Proofs of partial knowledge and simplified design of witness hiding protocols", Advances in Cryptology – CRYPTO'94, LNCS 839, pp. 174-187, 1994.

Appendix

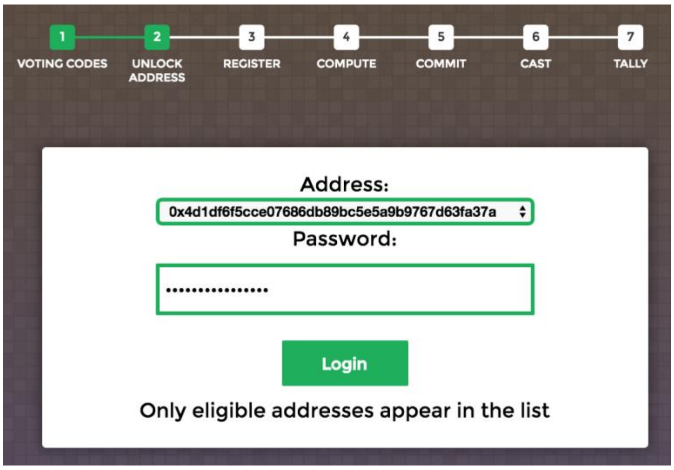

Figure 7: Voter page, Login to the Ethereum

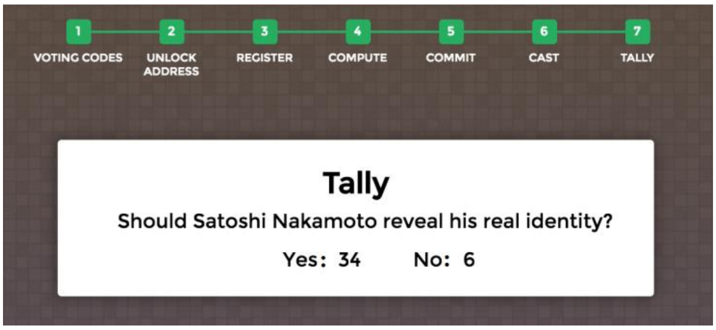

Figure 8: Voter Page, The election results

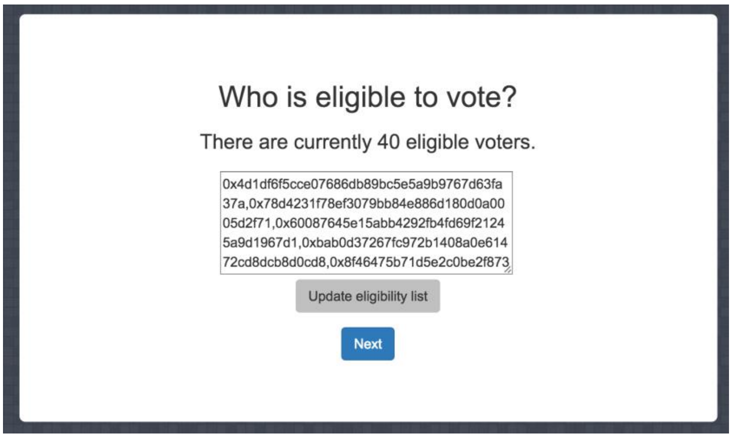

Figure 9: Election Authority page: Setting up the list of eligible voters

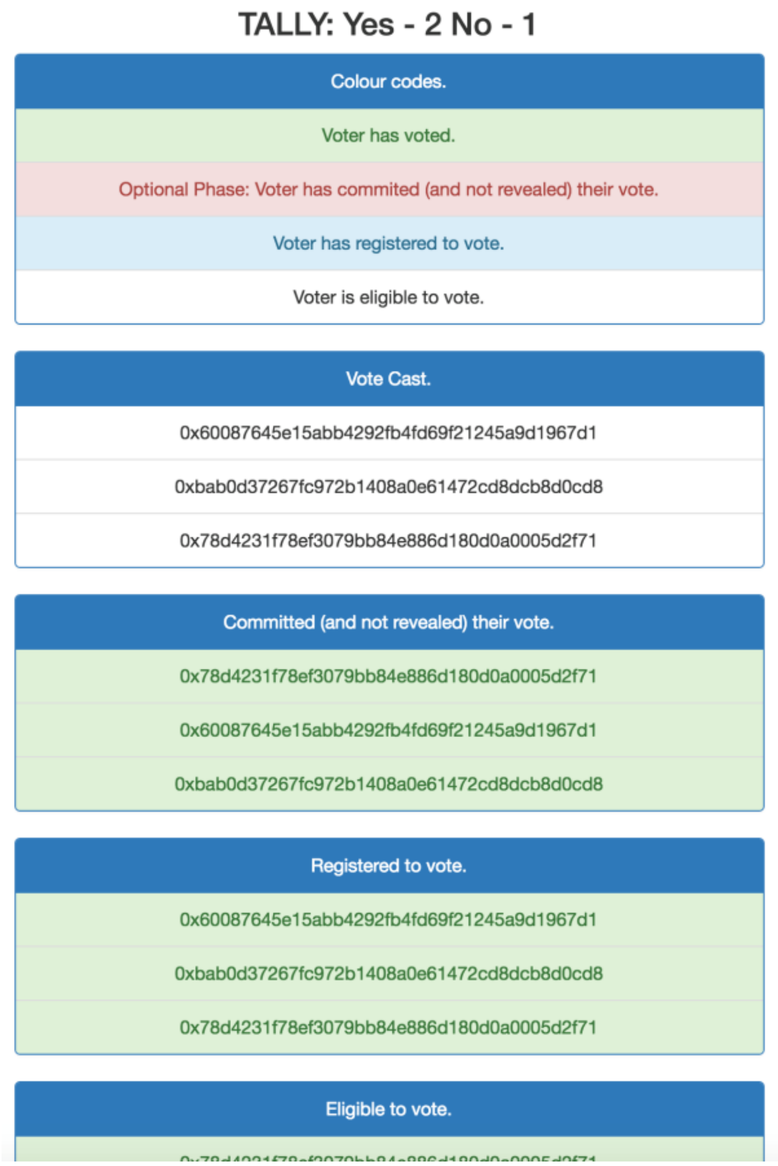

Figure 10: The Live Feed page showing voting in progress

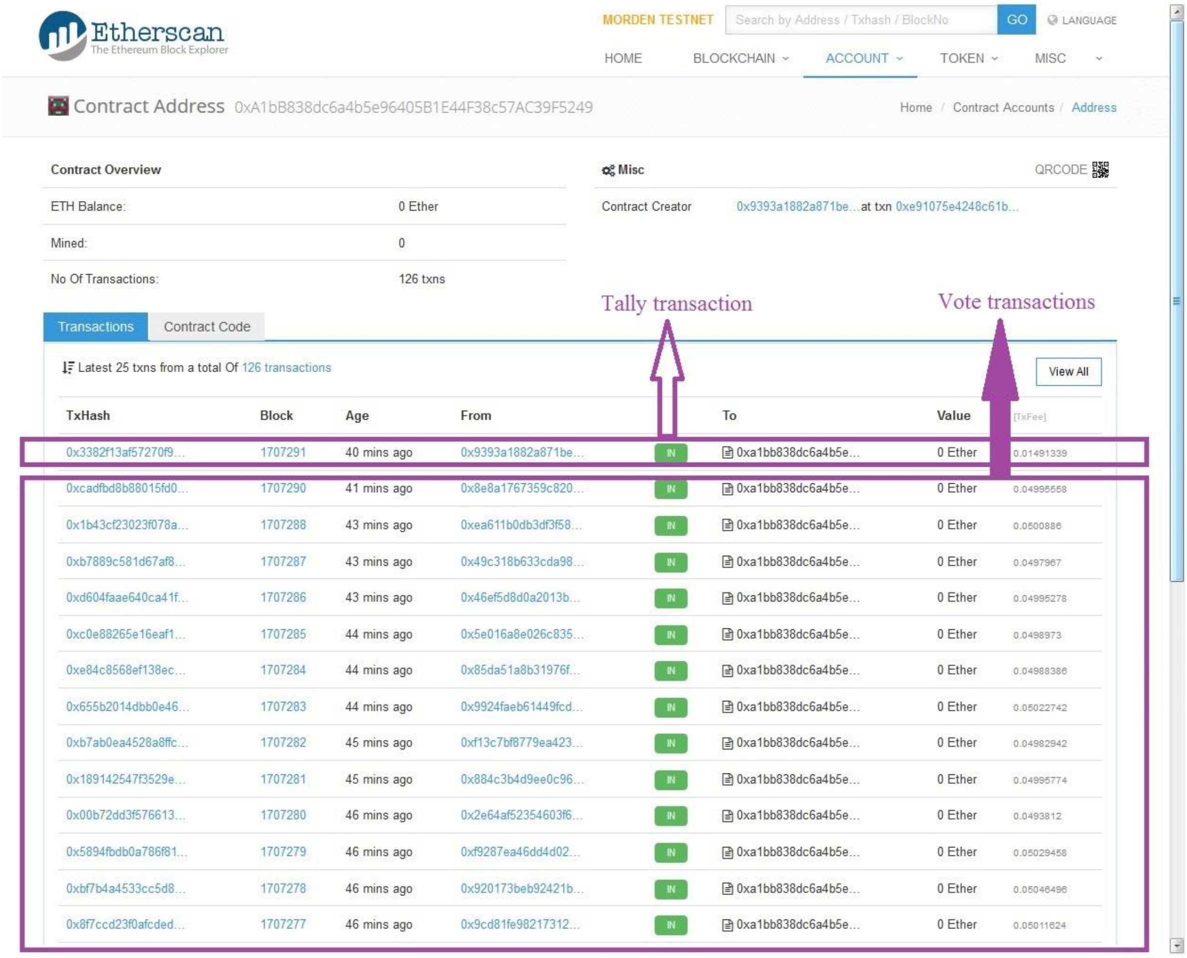

Figure 11: The transaction page of a sample election over Ethereum (https://testnet.etherscan.io/address/0xa1bb838dc6a4b5e96405b1e44f38c57ac39f5249)

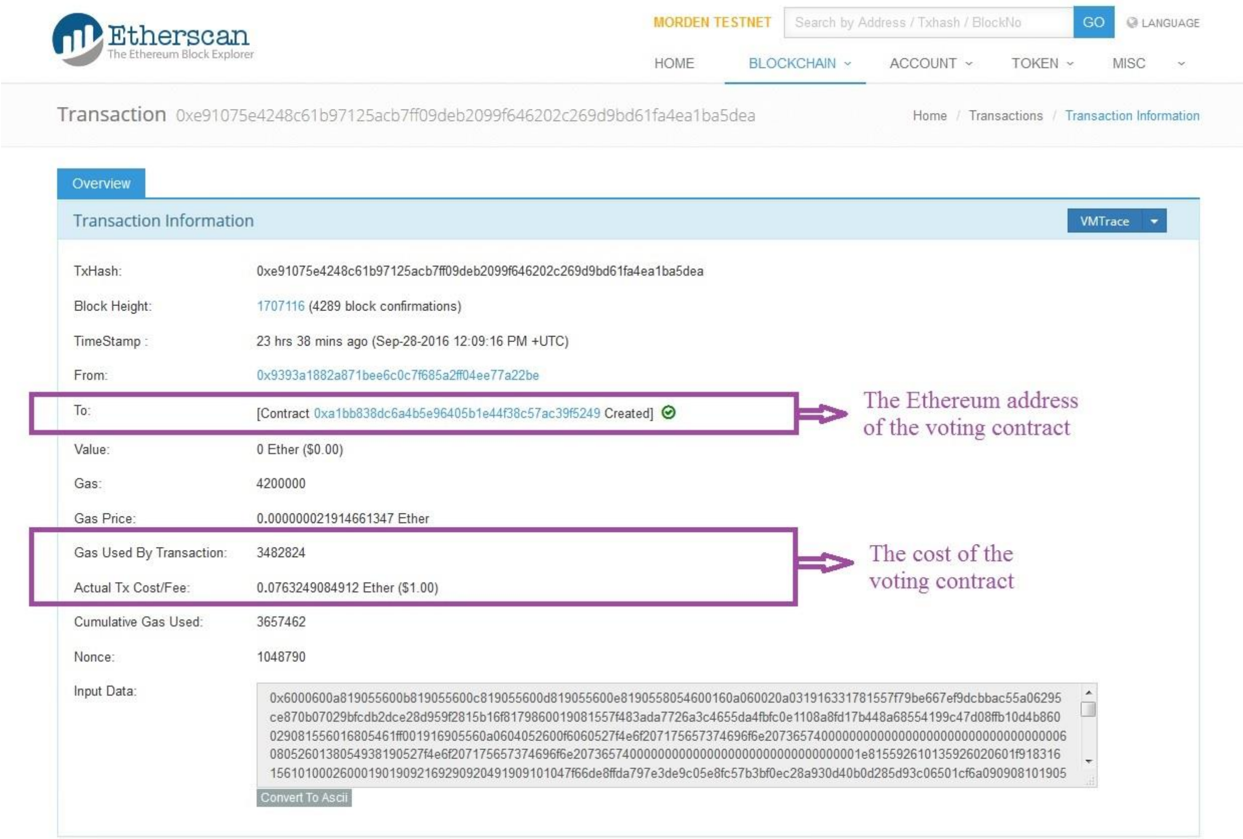

Figure 12: The transaction information of voting contract (https://testnet.etherscan.io/tx/0xe91075e4248c61b97125acb7ff09deb2099f646202c269d9bd61fa4ea1ba5dea)

Table 3: Cost of 40 voters participating in the Open Vote Network

Stages	**Gas per voter**	**Total Gas**	**Ether per voter**	**Total Ether**	**Cost per voter (£)**	**Total Cost (£)**
Eligible	53,840	2,153,610	0.0010768	0.0430722	0.010768	0.430722
Transition to SIGNUP	0	198,629	0	0.00397258	0	0.0397258
Register	743,323	29,732,914	0.01486646	0.59465828	0.1486646	5.9465828
Transition to COMPUTE	0	27,162	0	0.00054324	0	0.0054324
Compute	77,479	3,099,151	0.00154958	0.06198302	0.0154958	0.6198302
Commit	70,121	2,804,850	0.00140242	0.056097	0.0140242	0.56097
Vote	2,496,704	99,868,143	0.04993408	1.99736286	0.4993408	19.9736286
Tally	18,243	729,709	0.000364855	0.01459418	0.00364854	0.1459418
Total	3,441,467	139,330,329	0.06882934	2.78660658	£0.69	£27.72

Table 4: Comparison of cost with existing voting systems

Location	**Type of Election**	**Cost per Registered Voter**	**Election Title/Year**
UK Wide [11]	Paper Based	£3.01	UK Referendum/2011
California [12]	Paper/DRE with paper trail[17]	$2.77	General Election/2014
Colorado [13]	Mail[17]	$6.04	General Election/2014
North Dakota [14]	Paper Based[17]	$4.30	General Election/2014
Wisconsin [15]	Paper/DRE with paper trail[17]	$19.10 ($3.19 if all registrants showed up)	General Election/2014
India [16]	Electronic Voting Machines (EMVs) [18]	17 INR (about $0.25)	General Election/2009
Open Vote Network	Decentralized internet voting	£0.67	Trial election over Ethereum/2016