Time, Privacy, Robustness, Accuracy: Trade-Offs for the Open Vote Network Protocol

Fatima-Ezzahra El Orche, Rémi Géraud-Stewart, Peter Rønne, Gergei Bana, David Naccache, Peter Y. A. Ryan, Marco Biroli, Megi Dervishi, Hugo Waltsburger

Time, Privacy, Robustness, Accuracy: Trade-Offs for the Open Vote Network Protocol

Fatima-Ezzahra El Orche[1,2], Rémi Géraud-Stewart[4][0000−0001−8719−1724], Peter B. Rønne[2,6][0000−0002−2785−8301], Gergei Bana[3], David Naccache[1], Peter Y.A. Ryan[2][0000−0002−1677−9034], Marco Biroli[1], Megi Dervishi[1], and Hugo Waltsburger[5]

[1] ENS, CNRS, PSL Research University, Paris, France
{fatimaezzahra.elorche,david.naccache}@ens.fr
[2] SnT, FSTC, University of Luxembourg
{fatima.elorche,peter.ryan}@uni.lu
[3] University of Missouri at Columbia, Columbia MO, USA
banag@missouri.edu
[4] QPSI, Qualcomm Inc., San Diego CA, USA
rgerauds@qti.qualcomm.com
[5] CentraleSupelec, Université Paris-Saclay, Gif-sur-Yvette, France
hugo.waltsburger@centralesupelec.fr
[6] Universite de Lorraine, Inria, CNRS, France
peter.roenne@gmail.com

Abstract. The open vote network (OV-Net, [10]) is a secure two-round multi-party protocol facilitating the computation of a sum of integer votes without revealing their individual values. This is done without a central authority trusted for privacy, and thus allows decentralised and anonymous decision-making efficiently. As such, it has also been implemented in other settings such as financial applications, see e.g. [15,17].

An inherent limitation of OV-Net is its lack of robustness against denial-of-service attacks, which occur when at least one of the voters participates in the first round of the protocol but (maliciously or accidentally) not in the second. Unfortunately, such a situation is likely to occur in any real-world implementation of the protocol with many participants. This could incur serious time delays from either waiting for the failing parties and perhaps having to perform extra protocol rounds with the remaining participants.

This paper provides a solution to this problem by extending OV-Net with mechanisms tolerating a number of unresponsive participants, the basic idea being to run several sub-elections in parallel. The price to pay is a carefully controlled privacy loss, an increase in computation, and a statistical loss in accuracy, which we demonstrate how to measure precisely.

Keywords: Multi-party computation, open vote network, denial of service, decentralised voting.

1 Introduction

Cryptographic voting protocols allow mutually-distrusting entities to verifiably compute a voting result without revealing more about the private vote inputs than the actual result. Most of these protocols involve a trusted authority responsible for running the election or tallying the results. However, there exist a number of so-called "boardroom" or "self-tallying" schemes that do away with the need for a central authority [13]. In such decentralised schemes, the election is an interactive protocol between the voters only and it can even be made one-round, i.e. non-interactive, in a public key setting [7]. Whether a centralised or decentralised protocol is better-suited to a given situation depends on practical and context-specific concerns such as whether the trusted authority assumption makes sense. Especially, the decentralised protocol can be used in settings where there is no natural trusted third party, e.g., a company surveying privacy-sensitive data of the customers.

The open vote network (OV-Net) is a self-tallying voting scheme proposed by Hao, Ryan and Zieliński [10]. Improving upon Hao and Zieliński's earlier AV-net [11,9], it is a 2-round protocol which makes it an appealing candidate for larger-scale elections.[7] One of OV-Net's limitations, according to Hao–Ryan–Zieliński, is that the protocol cannot handle denial-of-service (DoS) events:

> *" (...) For example, if some voters refuse to send data in round 2, the tallying process will fail. This kind of attack is overt; everyone will know who the attackers are. To rectify this, voters need to expel the disrupters and restart the protocol; their privacy remains intact. However the voting process would be delayed, which may prove costly for large scale (countrywide) elections (...)"* — [10, Sec 3.4]

While the protection of privacy and the identification of culprits are desirable properties, the need to restart the protocol every time a voter drops out is a very strong limitation. This weakness is what we set out to rectify in this paper, by extending OV-Net to handle DoS events gracefully using parallel elections. Our modifications come at a cost, which we investigate quantitatively.

Some earlier works have already tried to improve the security and efficiency of OV-Net. In [12] fairness (i.e. preventing that voters get partial results before casting their vote) was guaranteed by committing to the vote in the first round. Further, the robustness against denial of service attacks was improved by introducing a recovery round: if some voters did not participate in the second round, the remaining voters perform a third round to achieve the partial tally for their cast votes. However this does not guarantee that there are no fallouts in the recovery round. In [7] it was shown that using a bilinear group setting and assuming a public key infrastructure, the voting protocol can be made non-interactive, i.e. one-round. This decreases the run time considerably, but does not in itself remove the robustness problem since the list of voters has to be

[7] As comparison, the self-tallying protocol of Groth [8] has $n + 1$ rounds for n voters, which makes it impractical to use for larger elections.

determined before the election and the result cannot be computed without every eligible voter participating. Finally, in [15] the OV-Net was implemented via a smart contract that financially punishes voters who drops out of the election. This gives an economic incentive to participate in the second round, but does not prevent dedicated DoS attacks, nor involuntary dropouts e.g. due to lack of network access, and it assumes that the participants are willing to risk the economic punishment in the first place.

2 Preliminaries

2.1 Notations

Throughout this paper, we will use the following notations. If X is a finite set, $x \xleftarrow{\$} X$ means that x is sampled uniformly at random from X. When working in a cyclic group $\mathbb{G}$ generated by g, we write $[x]$ to denote g^x. If $q > 1$ is an integer, we denote by $\mathbb{Z}_q := \mathbb{Z}/q\mathbb{Z}$ the ring of integers modulo q. We denote by $\mathbf{1}$ the vector whose coordinates are all 1. $\mathrm{BD}(p, n)$ denotes the binomial distribution of mean p for a population n.

Note that due to the page limit a longer version of paper including proofs of the obtained results and appendices can be accessed here [1].

2.2 Open vote network (OV-Net)

We recall here the OV-Net protocol in the simple case of a *referendum*: there are two vote choices encoded as 0 or 1 and n voters; each voter will cast a vote $v_i \in \{0, 1\}$ and the final tally will reveal the sum of all votes. Ultimately, we may set a threshold to choose a final winner based on the tally, but this is beyond the scope of OV-Net.

We assume that all participants have agreed ahead of time to use a given cyclic group $\mathbb{G}$ of generator g in which the decisional Diffie–Hellman problem is intractable. Let q be the order of $\mathbb{G}$. Each voter $i \in \{1, \dots, n\}$ samples a random value $x_i \xleftarrow{\$} \mathbb{Z}_q$ as a secret.

1. **Round 1:** Each voter $i \in \{1, \dots, n\}$ publishes g^{x_i} along with a zero-knowledge proof of knowledge (ZKP) of x_i, e.g. a Schnorr proof [16].
   When this round finishes, each voter $i \in \{1, \dots, n\}$ does the following:
   - checks the validity of the ZKP for all g^{x_j}, $j \in \{1, \dots, n\} \backslash \{i\}$,
   - computes: $g^{y_i} = \prod_{j=1}^{i-1} g^{x_j} / \prod_{j=i+1}^{n} g^{x_j}$
2. **Round 2:** Each participant $i \in \{1, \dots, n\}$ publishes $g^{x_i y_i} g^{v_i}$ and a ZKP for v_i showing that $v_i \in \{0, 1\}$. In practice, this proof can be implemented using the technique of Cramer–Damgård–Schoenmakers [5].

At the end of this procedure, each voter checks the proof of knowledge of all others, and multiplies together all the $g^{x_i y_i} g^{v_i}$'s. Since $\sum_i x_i y_i = 0$ by the definition of y_i, the result is $g^{\sum_{i=1}^n v_i}$, from which the value $\sum_{i=1}^n v_i$ can be

recovered by solving the discrete logarithm problem in G — this is tractable because n is small (by cryptographic standards), with the total world population being less than 2^{34}. Thus generic algorithms such as Pollard's ρ, with a complexity of $O(\sqrt{q})$, can be used here.

Remark 1. The OV-Net protocol can be extended to more than two candidates by an appropriate encoding of v_i [6,2], with the final tally requiring a (superincreasing) knapsack resolution after a discrete logarithm computation [10, Sec. 2.2]. Here we focus on the simpler case of two candidates.

2.3 Denial of Service

In the description of OV-Net, we implicitly assume that all participants are honest, to the extent that the proofs of knowledge are valid and that they follow the protocol. If one or several voters publish an incorrect proof of knowledge, or do not follow through with the protocol, then it is impossible to reach a conclusion for this particular vote event. This is called a denial of service (DoS) event.

When a DoS event occurs, the non-compliant voters can be identified and removed from a subsequent vote. However the results for that particular vote must be discarded (or cannot be computed) and a fresh vote must take place. This is troublesome for several reasons. One reason is that as n becomes large, disconnection or time-out events become more common and therefore the protocol's failure probability increases. Another reason is that accounting for protocol errors and re-voting adds complexity to real-world OV-Net implementations.

3 Parallel OV-Net

We consider a modification of OV-Net where users participate in several voting sessions in parallel. However, not all voters take part to all votes, as we now explain. Let n be the number of voters and M the number of parallel vote sessions. Each voter will participate in k pseudo-randomly chosen sessions amongst M.

More precisely, voter i picks k sessions before the protocol is run which we call i's *selection*. We assume that this selection is pseudo-random, i.e. that any given selection happens with the same probability $1/\binom{M}{k}$. As a result not all sessions have the same number of voters, a phenomenon that we will need to account for.

Remark 2. A natural question is whether we could impose a more clever rule, that would guarantee that there is always the same number of voting opportunities for each of them. Indeed, a solution is provided, in some cases, by Steiner systems [3]: a Steiner system with parameters t, k, n, written $S(t, k, n)$, is an n-element set S together with a set of k-element subsets of S (called *blocks*) with the property that each t-element subset of S is contained in exactly *one* block.

The existence of Steiner systems is deeply connected to number-theoretic properties, and in particular the existence of a $S(t, k, n+1)$ system precludes that of a $S(t, k, n)$. Thus, although we could initially form a balanced set of

voters in some initial setting, it cannot be done if any of the voters bails out (or is disconnected).

However, it is not obvious how a decentralised pool of voters could agree on such a setting in a non mutually-trusting way and without leaking private information. It also remains an interesting question whether approximately balanced block designs exist that are "stable" in the sense that they retain this property when elements are removed. ◇

Should a voter drop out during a voting session, this particular session will be discarded, but all sessions in which this voter didn't participate will go through. Unfortunately, this also discards all the votes of honest voters in the dropped session. To overcome this exclusion we allow each voter to vote k times: in other words, each voter will cast k votes into k independent ballots amongst the M.

Our claim is that in this case, the final tally's result reflects the choice of honest voters even after discarding all the sessions that were blocked by a dishonest voter. Furthermore, when several voters are dishonest, their cumulative effect on the final tally is weighed down by the fact that they shared many vote sessions. Concretely, for $k = M/2$, the first dishonest voter makes about $M/2$ sessions invalid; but amongst the remaining sessions only about $M/4$ can share a second dishonest voter, etc. Hence, this setting tolerates roughly $\log_2 M$ dropouts, at the price of running M sessions.

In summary, by running several sessions, several competing phenomena occur:

1. The overall protocol's resilience against DoS events is improved as we run more sessions — more sessions however bring an additional computational and communication cost;
2. Sessions have a varying number of voters in them, and not every voter partakes in every session, which introduces a bias — we can expect this bias to become small when many sessions are run;
3. The list of participants in each session is public, therefore some information about individual voters' preferences is leaked — running more sessions results in a increased loss of privacy.

There is therefore a balance to be struck, and we must quantify these phenomena more precisely.

4 Parallel OV-Net DoS resilience

Let ℓ be the number of voters causing a DoS event; they cause a (random) number X_ℓ of sessions to be discarded. The protocol fails when all sessions have been discarded, i.e., when $X_\ell \geq M$ — this cannot happen when $\ell < M/k$. If $\ell \geq M/k$ then it is possible to stop the protocol entirely when the selections of dropping voters cover all sessions. However, the likelihood of this happening when each selection is random and independent is low, as many of the dropping voters will have sessions in common.

This is a particular variant of the famous coupon collector's problem, which has been extensively studied.

Lemma 1. *The average number of DoS events necessary to cause an overall failure, when we run M parallel sessions and each voter partakes in k of them is*

$$\mathbb{E}[\ell \mid \textit{overall protocol failure}] = \binom{M}{k} \sum_{r=1}^{M} (-1)^{r-1} \frac{\binom{M}{r}}{\binom{M}{k} - \binom{M-r}{k}}.$$

Proof. See Appendix A.1 in [1].

Figure 1 compares simulation results to the formula of Lemma 1, showing excellent agreement. The simulation is for $M = 50$ and k varying from 1 to 49, over 10^5 runs[8]. Using this information, we can choose parameters M and k to accommodate a given number of potential drop-outs.

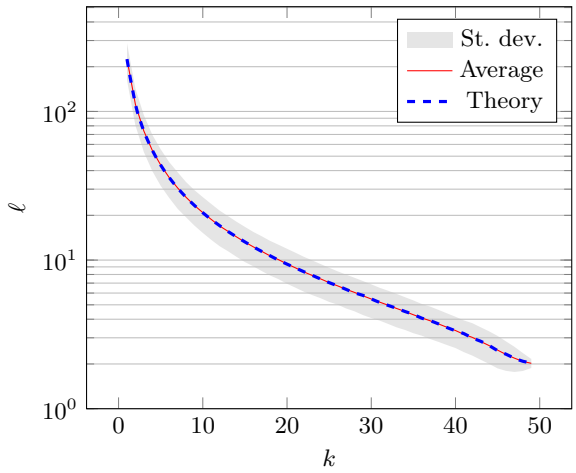

Fig. 1. Simulated and predicted minimum number of DoS events necessary to cause an overall protocol failure, for $M = 50$ and $k = 1, 2, \ldots, 49$.

When we have fewer than the critical number of DoS events, the remaining sessions can be tallied. We can estimate the number of remaining valid sessions as $\mu = M - X_\ell$:

Lemma 2. $\mathbb{E}(\mu) = (M - k)\left(1 - \frac{k}{M}\right)^{\ell - 1}$

Proof. See Appendix A.2 in [1]

Finer results about the distribution X_ℓ are given in Appendix A.5 in [1].

[8] The corresponding Python code is available from the authors upon request.

5 Tally-combining algorithms

In this section we formalise how a final result can be obtained from the parallel OV-Net protocol. It is practical at this point to use vector notations.

We make the assumptions that voters are consistent, i.e., that they make the same choice across all the voting sessions in which they participate[9]. We denote v_i the choice of voter i, and collect this (unknown) information into a vector $\boldsymbol{v} = (v_1, \ldots, v_n)$. If the vote went through with no incident, we would obtain the final tally : $V = \sum_{i=1}^{n} v_i = \boldsymbol{v} \cdot \mathbf{1}$.

When a voter drops out, all the sessions in which he participated are discarded. Let $0 < \mu \leq M$ be the number of remaining sessions and for each session $j \in \{1, \ldots, \mu\}$ let $s_{j,i}$ be the number of times that voter i participated in session j; hence $s_{j,i}$ can take values in $\{0, 1\}$ with the minimum value meaning that voter i did not partake in session j, and the maximum value indicating that they voted during session j. The tally for session j is therefore $t_j := \sum_{i=1}^{n} s_{j,i} v_i = \boldsymbol{v} \cdot \boldsymbol{s}_j$ where $\boldsymbol{s}_j := (s_{j,1}, \ldots, s_{j,n})$. By definition, $s_{j,i} = 0$ if voter i dropped out, and $\boldsymbol{s}_j$ is non-zero (otherwise $\mu = 0$). At the end of the procedure, the following information is public knowledge: $\boldsymbol{T} := (t_1, \ldots, t_\mu) \quad \boldsymbol{S} := (\boldsymbol{s}_1, \ldots, \boldsymbol{s}_\mu)$

The question is now: given $(\boldsymbol{S}, \boldsymbol{T})$, and the parameters $\mathsf{pp} = (n, k, M, \mu)$ how well can we approximate V? To answer this question we need a precise definition of the error.

Definition 1 (Average- and worst-case error). *Let $\mathcal{A}$ be an algorithm taking as input $\boldsymbol{S}$, $\boldsymbol{T}$ and (implicitly)* pp*, and returning a real number. We refer to $\mathcal{A}$ as a* tally-combining algorithm*, and we write $\delta(\boldsymbol{v}, \boldsymbol{S}) := V - \mathcal{A}(\boldsymbol{S}, \boldsymbol{T})$ for the* tallying error*.*

Since δ depends on a choice of $\boldsymbol{v}$, which is not public information, and since $\boldsymbol{S}$ is a collection of randomly chosen selections, it is more meaningful to consider the average error*:*

$$\pi_{avg}^{\mathcal{A}} := \mathbb{E}_{\boldsymbol{v},\boldsymbol{S}}[\delta(\boldsymbol{v}, \boldsymbol{S})],$$

where $\boldsymbol{v}$ and $\boldsymbol{S}$ span all their possible values.

While $\mathcal{A}$ may give results that are close to V on average, there may be corner cases in which the predicted value wanders substantially away from V; this phenomenon is controlled by the worst-case error*:*

$$\pi_{wc}^{\mathcal{A}} := \max_{\boldsymbol{v},\boldsymbol{S}} |\delta(\boldsymbol{v}, \boldsymbol{S})|,$$

where again $\boldsymbol{v}$ and $\boldsymbol{S}$ span all their possible values.

A simple tally-combining algorithm is given by averaging the tallies and rescaling to account for lost sessions, i.e.

$$\mathcal{A}_{\text{naïve}}(-, T) = \frac{M}{\mu k}(\mathbf{1} \cdot \boldsymbol{T})$$

[9] This makes our analysis simpler, but in practice a voter casting inconsistent votes simply weakens his own position.

(we must divide by k since each voter casts k votes).

Lemma 3. *The naïve tally-combining algorithm gives:*

$$\pi_{avg}^{naïve} = 0$$

Proof. See Appendix A in [1].

See also [1] for the worst case values.

More generally, let $\boldsymbol{x} = (x_1, \ldots, x_\mu)$ be a vector of real coefficients, and define the *weighed tally-combining algorithm* $\mathcal{A}_{\boldsymbol{x}}(T) = \boldsymbol{x} \cdot \boldsymbol{T}$, which gives the result

$$V_{\boldsymbol{x}} = \boldsymbol{x} \cdot \boldsymbol{T} = \boldsymbol{v} \cdot \left(\sum_{j=1}^{\mu} x_j \boldsymbol{s}_j \right) = \boldsymbol{v} \cdot \boldsymbol{\beta}_{\boldsymbol{x}}.$$

How do we choose $\boldsymbol{x}$? The following result partially answers this question

Theorem 1. *A sufficient condition for the bias of $\mathcal{A}_{\boldsymbol{x}}$ to be zero in average is $\mathbf{1} \cdot (\mathbf{1} - \boldsymbol{w}) = 0$ where $\boldsymbol{w} = x_1 \boldsymbol{s}_1 + \cdots + x_\mu \boldsymbol{s}_\mu$. Furthermore, under these conditions, standard deviation is proportional to $\|\mathbf{1} - \boldsymbol{w}\|_2^2$.*

Proof. See Appendix A.4 in [1].

If $\boldsymbol{S}$ spans $\mathbb{R}^n$, then by definition of a generating family we can find $\{x_1, \ldots, x_\mu\}$ such that $\boldsymbol{w} = \mathbf{1}$.[10] Concretely, we can construct an orthonormal basis of $\mathbb{R}^n$ from vectors of $\boldsymbol{S}$ and project $\mathbf{1}$ onto each coordinate. We dub this method of computing $\boldsymbol{x}$ the *minimum variance tally-combining algorithm* (MV, Table 1). When $\boldsymbol{S}$ span $\mathbb{R}^n$, the MV algorithm gives an exact result (zero bias and variance).

Input: $\boldsymbol{S} = \{\boldsymbol{s}_j\}$, $\boldsymbol{T}$, μ, n
Output: $V_{\boldsymbol{x}}$, $\boldsymbol{x}$, $\boldsymbol{w}$

1. $Z \leftarrow \emptyset$
2. For each $\boldsymbol{s}_j \in \boldsymbol{S}$, if $\boldsymbol{s}_j$ is linearly independent from Z, $Z \leftarrow Z \cup \boldsymbol{s}_j$
3. $\widehat{Z} \leftarrow \text{GramSchmidtOrthogonalisation}(Z)$
4. For each $\hat{\boldsymbol{z}}_j$, let $\hat{x}_j \leftarrow \mathbf{1} \cdot \hat{\boldsymbol{z}}_j$
5. $\boldsymbol{w} \leftarrow \sum_j \hat{x}_j \cdot \hat{\boldsymbol{z}}_j$
6. $M \leftarrow (\boldsymbol{z}_j \cdot \hat{\boldsymbol{z}}_\ell)_{j,\ell}$
7. $\boldsymbol{x} \leftarrow (M^\top)^{-1} \cdot \boldsymbol{w}$
8. $V_x \leftarrow \boldsymbol{x} \cdot \boldsymbol{T}$
9. Return $V_x, \boldsymbol{x}, \boldsymbol{w}$

Table 1. Algorithm for minimum variance tally combining (MV).

[10] The average value of μ such that $\boldsymbol{S}$ spans $\mathbb{R}^n$ is $\sum_{k=1}^{n} \frac{2^k}{2^k - 1}$. See [4] for more precise results.

However, when $\boldsymbol{S}$ does not span $\mathbb{R}^n$, the MV algorithm can only find a vector $\boldsymbol{w}$ close to $\mathbf{1}$, namely the closest such vector in terms of Euclidean distance that can be expressed in terms of vectors in $\boldsymbol{S}$. This is still the solution resulting in the smallest variance, but no longer the solution with the least bias!

This leads us to consider the following approach: we can construct tally-combining algorithms that guarantee zero bias, and select amongst these an algorithm that minimizes variance. Indeed, the constraint $\mathbf{1} \cdot (\mathbf{1} - \boldsymbol{w}) = 0$ can be guaranteed by determining x_1 as a linear function of other variables[11]. It remains to minimize $\|\mathbf{1} - \boldsymbol{w}\|_2^2$ which is simply a quadratic form in $\mu - 1$ variables. Therefore its minimum is easy to find as it amounts to solving a linear system in $\mu - 1$ rational variables. We call the corresponding algorithm the *zero-bias minimum variance tally-combining algorithm* (ZBMV, Table 2). In table 2, "symbolic expression" refers to the notion that $x_1, \ldots, x_\mu$ are not evaluated but are symbols to be manipulated formally.

Input: $\boldsymbol{S} = \{\boldsymbol{s}_j\}$, $\boldsymbol{T}$, μ, n
Output: $V_{\boldsymbol{x}}$, $\boldsymbol{x}$

1. Let x_1 be the symbolic expression $\frac{1}{\mathbf{1} \cdot \boldsymbol{s}_1} \left(n - \sum_{j=2}^{\mu} x_j (\mathbf{1} \cdot \boldsymbol{s}_j) \right)$
2. Let D be the symbolic expression $\|\mathbf{1} - \sum_{j=1}^{\mu} x_j \boldsymbol{s}_j\|_2^2$
3. $(x_2^\star, \ldots, x_\mu^\star) \leftarrow$ solutions of the linear system $\nabla D = 0$
4. $x_1^\star \leftarrow \frac{1}{\mathbf{1} \cdot \boldsymbol{s}_1} \left(n - \sum_{j=2}^{\mu} x_j^\star (\mathbf{1} \cdot \boldsymbol{s}_j) \right)$
5. $\boldsymbol{x} \leftarrow (x_1^\star, \ldots, x_\mu^\star)$
6. $V_x \leftarrow \boldsymbol{x} \cdot \boldsymbol{T}$
7. Return $V_x, \boldsymbol{x}$

Table 2. Algorithm for zero-bias minimum variance tally combining.

5.1 Comparing tally-combining algorithms

Let's consider a toy example to illustrate how the three discussed tally-combining algorithms compare. Throughout this section, we take $n = 4$, $M = 6$, $\mu = 3$, $k = 3$ and $\boldsymbol{s}_1 = (1, 1, 1, 0)$, $\boldsymbol{s}_2 = (1, 1, 0, 0)$, $\boldsymbol{s}_3 = (0, 1, 0, 1)$ and $\boldsymbol{T} = (1, 0, 0)$.[12] The results are summarized in Table 3.

Algorithm 1 (Zero-bias minimum variance) *We can express x_1 in terms of x_2 and x_3 to ensure zero bias:*

$$x_1 = \frac{1}{\mathbf{1} \cdot \boldsymbol{s}_1} (n - x_2 (\mathbf{1} \cdot s_2) - x_3 (\mathbf{1} \cdot s_3)) = \frac{1}{3} (4 - 2x_2 - 2x_3) .$$

[11] There is nothing special about $\boldsymbol{s}_1$, any other vector of $\boldsymbol{S}$ can be used. Note that $\mathbf{1} \cdot \boldsymbol{s}_1 \neq 0$.

[12] Note that in this example, knowing the tallies t_1 and t_2 reveals one participant's vote. This privacy issue is addressed later in the paper.

Tally-combining algorithm	Bias $\mathbf{1}\cdot(\mathbf{1}-\boldsymbol{w})$	Variance $\|\mathbf{1}-\boldsymbol{w}\|_2^2$	Tally $\boldsymbol{x}\cdot\boldsymbol{T}$
Naïve algorithm	-2/3	4/3	2/3
ZBMV	0	5/7	6/7
MV	1/3	1/3	1

Table 3. Comparison between tally-combining algorithms on the toy example.

We are left to determine x_2 and x_3, which we choose to minimize the distance of $\boldsymbol{w} = x_1\boldsymbol{s}_1 + \cdots + x_3\boldsymbol{s}_3$ to $\mathbf{1}$, i.e. the quantity

$$\|\mathbf{1}-\boldsymbol{w}\|_2^2 = \sum_{i=1}^{n}(1-w_i)^2 = (1-x_1-x_2)^2 + (1-x_1-x_2-x_3)^2 + (1-x_1)^2 + (1-x_3)^2$$
$$= \frac{1}{3}(4 + 5x_2^2 + 2x_2(x_3-3) + 3x_3^2 - 2x_3)$$

This achieves its global minimum value of $5/7$ at $x_2^\star = 4/7$ and $x_3^\star = 1/7$. Therefore, we have: $\boldsymbol{x} = \frac{1}{7}(6,4,1)$. In particular, $\boldsymbol{w} = x_1^\star\boldsymbol{s}_1 + \cdots + x_3^\star\boldsymbol{s}_3 = \frac{1}{7}(10,11,6,1)$ (note that computing this vector is not necessary for the algorithm).

Algorithm 2 (Minimum variance) *We begin by computing an orthonormal basis $\hat{Z}$ from $\boldsymbol{S}$: $\hat{z}_1 = \frac{1}{\sqrt{3}}(1,1,0,0)$ $\hat{z}_2 = \left(\frac{1}{\sqrt{6}}, \frac{1}{\sqrt{6}}, -\sqrt{\frac{2}{3}}, 0\right)$ $\hat{z}_3 = (-\frac{1}{\sqrt{6}}, \frac{1}{\sqrt{6}}, 0, \sqrt{\frac{2}{3}})$ which gives $\hat{x}_1 = \sqrt{3}$, $\hat{x}_2 = 0$, $\hat{x}_3 = \sqrt{2/3}$, from which we get $\boldsymbol{w} = \frac{1}{3}(2,4,3,2)$ and finally $\boldsymbol{x} = \left(1, -\frac{1}{3}, \frac{2}{3}\right)$.*

As expected this tally-combining algorithm has smaller variance (since $\|\mathbf{1}-\boldsymbol{w}\|_2^2 = 1/3$), compared with the ZBMV algorithm in of Algorithm 1, but its bias is not guaranteed to be zero (since $\mathbf{1}\cdot(\mathbf{1}-\boldsymbol{w}) = 1/3$).

Algorithm 3 (Naïve tally combining) *Let's use the naïve tally-combining algorithm, i.e., $\boldsymbol{x} = \frac{M}{\mu k}\mathbf{1}$. We assume here that $M = 6$, $\mu = 3$ and $k = 3$ so that $\boldsymbol{x} = \frac{2}{3}\mathbf{1}$, yielding $\boldsymbol{w} = (\frac{4}{3}, 2, \frac{2}{3}, \frac{2}{3})$. The bias for this algorithm is $-2/3$, however this algorithm has larger variance than the other two, since $\|1-\boldsymbol{w}\|_2^2 = 4/3$.*

6 Privacy of Parallel OV-Net

In this section we investigate the decrease in privacy which we can expect due to the multiple parallel elections which are tallied individually, thus giving the adversary extra information. As an example, let us consider a simple referendum. If the outcome is unanimous , we of course lose privacy. However, the probability of this might be small. However, if we split the voters into two elections, the probability is roughly the square root of the old probability, i.e. much higher.

Recall that M is the number of the parallel and independent elections, n is the total number of voters and k is the number of elections that each voter has randomly chosen to participate in. We denote by M_i the set of voters who

participated in election i and we consider that the elections are enumerated from 1 to M. Let $\mathrm{Res}(M_i)$ be the random variable that gives the number of 'Yes' votes in the set M_i. We recall also that Y_i is the random variable that gives the number of voters in the set M_i.

6.1 Definitions and Assumptions

To quantify privacy, we use the δ-privacy definition for voting from [14] which assumes that, besides the voting elements of a voting protocol, there exists an additional party called an observer O, who can observe publicly available information. Moreover, we assume that among the n honest voters, there exists a voter V_{obs} who is under observation. For the sake of clarity, V_{obs} will refer at the same time to the voter under observation and to its vote.

Definition 2. *Let P be a voting protocol and V_{obs} be the voter under observation. We say that P achieves δ-privacy if the difference between the probabilities*

$$\mathbb{P}[(\pi_O||\pi_{V_{obs}}(Yes)||\pi_v)^{(l)} \to 1] \text{ and } \mathbb{P}[(\pi_O||\pi_{V_{obs}}(No)||\pi_v)^{(l)} \to 1]$$

is δ-bounded as a function of the security parameter ℓ, where π_O, $\pi_{V_{obs}}$ and π_v are respectively the programs run by the observer O, the voter under observation V_{obs} and all the honest voters v (clearly without V_{obs}).

To calculate the privacy we use the following result from [14]

$$\delta(n) = \sum_{r \in M^*_{\mathrm{Yes,No}}} (A_r^{\mathrm{No}} - A_r^{\mathrm{Yes}}) \tag{1}$$

where $M^*_{\mathrm{Yes,No}} = \{r \in \mathbb{R} : A_r^{\mathrm{Yes}} \leq A_r^{\mathrm{No}}\}$, $\mathbb{R}$ is the set of all possible election results and A_r^j denotes the probability that the choices of the honest voters yield the result r of the election given that V_{obs}'s choice is j.

We consider a referendum with n honest voters with a uniform distribution between yes and no votes. For simplicity, we will assume that nobody abstains. We also assume that no voters are corrupted. This is reasonable, since instructing corrupted voters to vote in a special way does not give further advantage compared to simply knowing the corrupted voters' votes. Moreover, we assume that at least one of the elections in which V_{obs} participated is surviving.

6.2 Basic Cases: $M = k = 1$ and $M \geq 1, k = 1$

The δ for a single referendum is :

$$\begin{aligned}
\delta(n) &= \left(\frac{1}{2}\right)^n \frac{1}{n} \sum_{a=0}^{n} \binom{n}{a} |2a - n| \\
&= \begin{cases} 2^{-n}\binom{n}{\frac{n}{2}} & \text{if } n \text{ is even} \\ \dfrac{2^{1-n}}{n}\binom{n}{1+\left[\frac{n}{2}\right]}\left(1 + \left[\dfrac{n}{2}\right]\right) & \text{Otherwise} \end{cases}
\end{aligned}$$

where the first equality holds using the result from (1) and the second one using the binomial theorem.

The formula above refers to the case $M = k = 1$ where all voters had chosen to vote in the same and unique election 1. For the case $M > 1$ and $k = 1$, δ becomes a random variable and the expected value of δ of the election in which V_{obs} is participating can be defined as follows:

$$\delta_{\text{expected}}(n, M) = \sum_{n'=1}^{n} \mathbb{P}(Y_i' = n')\delta(n') \tag{2}$$

where Y_i' is the random variable that gives the number of voters who participated in the election i, including V_{obs}; and $Y_i' \sim 1 + \text{BD}(n-1, \frac{k}{M})$. Equation (2) for $k = 1$ and $M > 1$ becomes:

$$\delta_{\text{expected}}(n, M) = \sum_{n'=1}^{n} \binom{n-1}{n'-1} \left(\frac{1}{M}\right)^{n'-1} \left(1 - \frac{1}{M}\right)^{n-n'} \delta(n')$$

Figure 2 shows that privacy is almost lost when $M \gg n$.

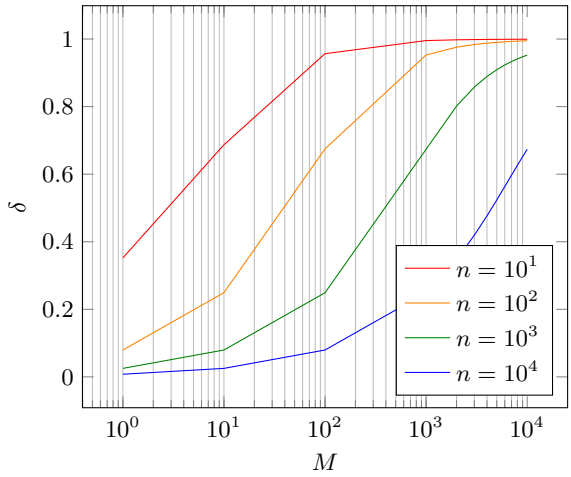

Fig. 2. The relationship between M and δ_{expected} for different values of $n = 10, 10^2, 10^3, 10^4$.

6.3 General Case

In this part we give a general formula of δ. To this end, we consider the following. Let $y = (y_1, \ldots, y_M)$ be an assignment of voters such that $\text{Card}(M_i) = y_i$ for $i \in [1, M]$. We can obtain all the possible assignments of voters by respecting the condition $\sum_{i=1}^{M} y_i = nk$. Let $r = (r_1, \cdots, r_M)$ be a possible result corresponding to the assignment y with $r_i = \text{Res}(M_i)$ for $i \in [1, M]$. r verifies the conditions $(\sum_{i=1}^{M} r_i) \bmod k = 0$ and $r_i \leq y_i$ for $i \in [1, M]$. Remember that $\text{Res}(M_i)$ gives

the number of "Yes" votes in M_i. We have $\text{Res}(M_i) \sim \text{BD}(y_i, \frac{1}{2})$ for $i \in [1, M]$. Intuitively, δ can be expressed as the following:

$$\delta(n, M, k) = \sum_{y_1+\cdots+y_M=nk} \mathbb{P}(Y_1 = y_1, \ldots, Y_M = y_M) \cdot \sum_{r \in M^*_{\text{Yes,No}}} (A_r^{\text{No}} - A_r^{\text{Yes}})$$

By definition of A_r^j we have $A_r^j = \mathbb{P}(\text{Res}(M_1) = r_1, \ldots, \text{Res}(M_M) = r_M / V_{\text{obs}} = j)$ with $j \in \{\text{Yes, No}\}$.

To proceed we will introduce an additional notation. Remember that M_i denotes the voters in election i. Define Σ_k as the subsets of $\{1, \ldots, M\}$ of cardinality k. For $\sigma \in \Sigma_k$ we define $M'_\sigma = \bigcap_{i \in \sigma} M_i$, i.e. the voters participating in the elections in the set σ. Note that the assignment of voters to elections is uniformly random, i.e. each voter is assigned uniformly and uniquely to a M'_σ. Also Z_σ is the random variable determining the number of voters in M'_σ.

There are $c = \binom{M}{k}$ possible M'_σs . Suppose that σs are enumerated from 1 to c. Let $z = (z_{\sigma_1}, \ldots, z_{\sigma_c})$ be an assignment of voters such that $z_{\sigma_i} = \text{Card}(M'_{\sigma_i})$, for $(\sigma_i, i) \in \Sigma_k \times [1, c]$. All the possible assignments of voters z are obtained by respecting the condition $\sum_{\sigma_i \in \Sigma_k} z_{\sigma_i} = n$.

The variables Z_σ, $\sigma \in \Sigma_k$ correspond to the problem of putting n indistinguishable balls into c distinguishable boxes, i.e. the vector $Z = (Z_{\sigma_1}, \ldots, Z_{\sigma_c})$ follows a multinomial distribution with equal parameters $p_i = 1/c$, and $\sum_{\sigma \in \Sigma} z_\sigma = n$ including V_{obs}. We can now calculate the probability for the assignment of the voters, and rewrite our formula as:

$$\delta(n, M, k) = \sum_{z_1+\cdots+z_c=n} \mathbb{P}(Z_{\sigma_1} = z_{\sigma_1}, \cdots, Z_{\sigma_c} = z_{\sigma_c}) \cdot \sum_{r \in M^*_{\text{Yes,No}}} (A_r^{\text{No}} - A_r^{\text{Yes}})$$

Let $r' = (r'_{\sigma_1}, \ldots, r'_{\sigma_c})$ such that $r'_{\sigma_i} = \text{Res}(M'_{\sigma_i})$ for $(\sigma_i, i) \in \Sigma_k \times [1, c]$. The variables $\text{Res}(M'_\sigma)$, $\sigma \in \Sigma_k$, are independent and follow the binomial distribution of parameters z_σ and $1/2$.

In the case $M = c$, which means $k = M - 1$ or $k = 1$, there is a one-to-one correspondence between the sets $(M_i)_{i \in [1,M]}$ and $(M'_\sigma)_{\sigma \in \Sigma_k}$. However this is not true in general and we have a relation between r and r' defined by the function f as follows:

$$\begin{pmatrix} r_1 \\ \vdots \\ r_M \end{pmatrix} = B \cdot \begin{pmatrix} r'_{\sigma_1} \\ \vdots \\ r'_{\sigma_c} \end{pmatrix} = f(r'_{\sigma_1}, \cdots, r'_{\sigma_c}) \text{ where } B = (b_{i\sigma})_{\substack{1 \le i \le M \\ \sigma \in \Sigma_k}} \text{ and } b_{i\sigma} = \mathbf{1}_{i \in \sigma}.$$

We can now calculate the probability A_r^{v} as: $A_r^{\text{v}} = \sum_{r' | r = f(r')} A'^{\text{v}}_{r'}$ and we have: $A'^{\text{v}}_{r'} = \mathbb{P}(\text{Res}(M'_{\sigma_1}) = r'_{\sigma_1}, \cdots, \text{Res}(M'_{\sigma_c}) = r'_{\sigma_c} / V_{\text{obs}} = \text{v})$

Suppose that V_{obs} is in the subset M'_{σ_1}. It is symmetric to choose any other subset. We have: $A'^{\text{v}}_{r'} = \left(\frac{1}{2}\right)^{z_{\sigma_1} - 1} \cdot h(z_{\sigma_1}, r'_{\sigma_1}) \cdot \prod_{i=2}^{c} \left(\frac{1}{2}\right)^{z_{\sigma_i}} \cdot \binom{z_{\sigma_i}}{r'_{\sigma_i}}$ where $h(x, y) =$

$$\begin{cases} \binom{x-1}{y-1} \text{ if v = "Yes"} \\ \binom{x-1}{y} \text{ if v = "No"} \end{cases}$$

Remember that: $M^*_{\text{Yes,No}} = \{r' : A'^{\text{Yes}}_{r'} \leq A'^{\text{No}}_{r'}\}$, and $A'^{\text{No}}_{r'} \geq A'^{\text{Yes}}_{r'}$ is true when $r'_{\sigma_1} \in [0, [\frac{z_{\sigma_1}}{2}]]$. We have $\sum_{r'_1=0}^{[\frac{z_{\sigma_1}}{2}]}(A'^{\text{No}}_{r'} - A'^{\text{Yes}}_{r'}) = \frac{1}{2}\sum_{r'_1=0}^{z_{\sigma_1}}|A'^{\text{No}}_{r'} - A'^{\text{Yes}}_{r'}|$.

Since V_{obs} is in M'_{σ_1}, the vector to consider is $Z' = (Z_{\sigma_1} - 1, Z_{\sigma_2}, \cdots, Z_{\sigma_c})$. The formula of δ becomes:

$$\delta(n, M, k) = a_n \cdot \sum_{z_{\sigma_1}=1}^{n} \frac{E(z_{\sigma_1})}{z_{\sigma_1}!} \sum_{z_{\sigma_2}=0}^{n} \cdots \sum_{z_{\sigma_c}=0}^{n} \frac{\delta_{\sum_{\sigma\in\Sigma} z_\sigma, n}}{z_{\sigma_2}! \cdots z_{\sigma_c}!} = a_n \cdot \sum_{z=1}^{n} \frac{E(z)}{z!} \cdot \frac{(c-1)^{n-z}}{(n-z)!}$$

with $a_n = \frac{(n-1)!}{c^{n-1}} \cdot \left(\frac{1}{2}\right)^n$, $E(z) = 2^{n-z+1}\binom{z}{[\frac{z}{2}]} \cdot \left[\frac{z}{2}\right]$ and $\delta_{i,j}$ is the Kroenecker delta function.

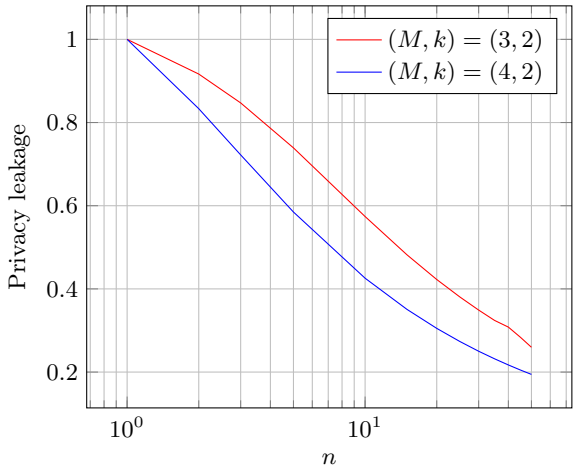

Fig. 3. Privacy leakage as function of n for the cases $(M, k) = (3, 2), (4, 2)$.

7 Conclusions and Further Research

Conclusions In this paper, we presented a new version of the protocol OV-Net which run several elections in parallel to achieve robustness against DoS failures without having to resort to time-consuming extra rounds. We computed quantitatively the increase in robustness from having M parallel elections with each voter participating in k of these, and demonstrated that robustness can be significantly improved. The improvement in time and robustness comes at a cost in terms of accuracy and privacy. We stress that our protocol is well fitting for decision-making applications where accuracy and privacy is not of ultimate importance. We presented three different algorithms on how to optimally compute the tally using this new OV-Net version and we quantitatively measured the

privacy decrease that is expected due to the multiple partial election results. The results allow the protocol initiator to choose parameters to carefully balance the wanted robustness with a controlled privacy loss, statistical loss in accuracy, as well as increased computation.

Future work An idea to consider is redistribution i.e. elections are conducted in several electoral districts. Unlike general elections, where the final result is known for the entire country only, in redistributed elections results are consolidated per district and only then added up. This could confine problematic voters to a district of their own, as follows: partition the n voters into d districts of $n' = n/d$ voters, then run a vote in each of them. Then recompose the result by adding up the final tally. This strategy confines the DoS problem to districts that do not influence each other. However, DoS tolerance is not exactly multiplied by d because each district is not allowed to exceed k unresponsive voters. In other words, tolerance is multiplied by d as long as the constraint that there are no more than k unresponsive voters per district is respected.

8 Acknowledgements

PYAR acknowledge support from the Luxembourg National Research Fund (FNR) under the CORE project EquiVox (C19/IS/13643617/EquiVox/Ryan) and FEEO was supported by the FNR grant PRIDE15/10621687/ SPsquared. This work received funding from the France 2030 program managed by the French National Research Agency under grant agreement No. ANR-22-PECY-0006.

References

1. Bana, G., Biroli, M., Dervishi, M., El Orche, F.E., Géraud-Stewart, R., Naccache, D., Rønne, P.B., Ryan, P.Y., Waltsburger, H.: Time, privacy, robustness, accuracy: Trade offs for the open vote network protocol. Cryptology ePrint Archive (2021)
2. Baudron, O., Fouque, P., Pointcheval, D., Stern, J., Poupard, G.: Practical multi-candidate election system. In: Kshemkalyani, A.D., Shavit, N. (eds.) Proceedings of the Twentieth Annual ACM Symposium on Principles of Distributed Computing, PODC 2001, Newport, Rhode Island, USA, August 26-29, 2001. pp. 274–283. ACM (2001). https://doi.org/10.1145/383962.384044
3. Colbourn, C.J., Dinitz, J.H.: Handbook of combinatorial designs. CRC press (2006)
4. Cooper, C., Frieze, A.M., Pegden, W.: On the rank of a random binary matrix. In: Chan, T.M. (ed.) Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019. pp. 946–955. SIAM (2019). https://doi.org/10.1137/1.9781611975482.58
5. Cramer, R., Damgård, I., Schoenmakers, B.: Proofs of partial knowledge and simplified design of witness hiding protocols. In: Desmedt, Y. (ed.) Advances in Cryptology - CRYPTO '94, 14th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1994, Proceedings. Lecture Notes in Computer Science, vol. 839, pp. 174–187. Springer (1994). https://doi.org/10.1007/3-540-48658-5_19

6. Cramer, R., Franklin, M.K., Schoenmakers, B., Yung, M.: Multi-autority secret-ballot elections with linear work. In: Maurer, U.M. (ed.) Advances in Cryptology - EUROCRYPT '96, International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996, Proceeding. Lecture Notes in Computer Science, vol. 1070, pp. 72–83. Springer (1996). https://doi.org/10.1007/3-540-68339-9_7
7. Giustolisi, R., Iovino, V., Rønne, P.B.: On the possibility of non-interactive e-voting in the public-key setting. In: International Conference on Financial Cryptography and Data Security. pp. 193–208. Springer (2016)
8. Groth, J.: Efficient maximal privacy in boardroom voting and anonymous broadcast. In: Juels, A. (ed.) Financial Cryptography, 8th International Conference, FC 2004, February 9-12, 2004. Revised Papers. Lecture Notes in Computer Science, vol. 3110, pp. 90–104. Springer (2004). https://doi.org/10.1007/978-3-540-27809-2_10
9. Hao, F.: A 2-round anonymous veto protocol. In: Christianson, B., Crispo, B., Malcolm, J.A., Roe, M. (eds.) Security Protocols, 14th International Workshop, Cambridge, UK, March 27-29, 2006, Revised Selected Papers. Lecture Notes in Computer Science, vol. 5087, pp. 212–214. Springer (2006). https://doi.org/10.1007/978-3-642-04904-0_29
10. Hao, F., Ryan, P.Y.A., Zielinski, P.: Anonymous voting by two-round public discussion. IET Information Security **4**(2), 62–67 (2010). https://doi.org/10.1049/iet-ifs.2008.0127
11. Hao, F., Zielinski, P.: A 2-round anonymous veto protocol. In: Christianson, B., Crispo, B., Malcolm, J.A., Roe, M. (eds.) Security Protocols, 14th International Workshop, Cambridge, UK, March 27-29, 2006, Revised Selected Papers. Lecture Notes in Computer Science, vol. 5087, pp. 202–211. Springer (2006). https://doi.org/10.1007/978-3-642-04904-0_28
12. Khader, D., Smyth, B., Ryan, P., Hao, F.: A fair and robust voting system by broadcast. Lecture Notes in Informatics (LNI), Proceedings-Series of the Gesellschaft fur Informatik (GI) pp. 285–299 (2012)
13. Kiayias, A., Yung, M.: Self-tallying elections and perfect ballot secrecy. In: Naccache, D., Paillier, P. (eds.) Public Key Cryptography, 5th International Workshop on Practice and Theory in Public Key Cryptosystems, PKC 2002, Paris, France, February 12-14, 2002, Proceedings. Lecture Notes in Computer Science, vol. 2274, pp. 141–158. Springer (2002). https://doi.org/10.1007/3-540-45664-3_10
14. Küsters, R., Liedtke, J., Müller, J., Rausch, D., Vogt, A.: Ordinos: A verifiable tally-hiding remote e-voting system. Tech. rep., Cryptology ePrint Archive (2020)
15. McCorry, P., Shahandashti, S.F., Hao, F.: A smart contract for boardroom voting with maximum voter privacy. In: Kiayias, A. (ed.) Financial Cryptography and Data Security - 21st International Conference, FC 2017, Sliema, Malta, April 3-7, 2017, Revised Selected Papers. Lecture Notes in Computer Science, vol. 10322, pp. 357–375. Springer (2017). https://doi.org/10.1007/978-3-319-70972-7_20
16. Schnorr, C.: Efficient signature generation by smart cards. J. Cryptol. **4**(3), 161–174 (1991). https://doi.org/10.1007/BF00196725
17. Seifelnasr, M., Galal, H.S., Youssef, A.M.: Scalable open-vote network on ethereum. In: Bernhard, M., Bracciali, A., Camp, L.J., Matsuo, S., Maurushat, A., Rønne, P.B., Sala, M. (eds.) Financial Cryptography and Data Security - FC 2020 International Workshops, AsiaUSEC, CoDeFi, VOTING, and WTSC, Kota Kinabalu, Malaysia, February 14, 2020, Revised Selected Papers. Lecture Notes in Computer Science, vol. 12063, pp. 436–450. Springer (2020). https://doi.org/10.1007/978-3-030-54455-3_31