Selene: Voting with Transparent Verifiability and Coercion-Mitigation

Peter Y.A. Ryan[1(✉)], Peter B. Rønne[1,2], and Vincenzo Iovino[1]

[1] University of Luxembourg, Esch-sur-Alzette, Luxembourg
peter.ryan@uni.lu, vinciovino@gmail.com
[2] INRIA Nancy, Villers-làs-Nancy, France
peter.roenne@inria.fr

Abstract. End-to-end verifiable voting schemes typically involve voters handling an encrypted ballot in order to confirm that their vote is accurately included in the tally. While this may be technically valid, from a public acceptance standpoint it may be problematic: many voters may not really understand the purpose of the encrypted ballot and the various checks that they can perform. In this paper we take a different approach and revisit an old idea: to provide each voter with a private tracking number. Votes are posted on a bulletin board in the clear along with their associated tracking number. This is appealing in that it provides voters with a very simple, intuitive way to verify their vote, in the clear. However, there are obvious drawbacks: we must ensure that no two voters are assigned the same tracker and we need to keep the link between voters and trackers private.

We propose a scheme that addresses both of these problems: we ensure that voters get unique trackers and we close off coercion opportunities by ensuring that the voters only learn their tracking numbers after the votes have been posted. The resulting scheme provides receipt-freeness, and indeed a good level of coercion-resistance while also providing a more immediately understandable form of verifiability.

1 Introduction

The challenge with voting systems is to provide sufficient evidence to render the outcome beyond dispute while at the same time ensuring ballot secrecy and coercion resistance. Furthermore, the system has to be very easy to use and easily understandable. The response from the crypto community has been to develop the notion of End-to-End (E2E) Verifiability. A number of schemes have been proposed and some even implemented and deployed, for example, Prêt à Voter [25] Wombat [2] and Scantegrity II [26], Helios https://vote.heliosvoting.org/, Civitas [7], Pretty Good Democracy [24].

Typically these schemes involve the creation of an encrypted version of the vote at the time of casting. The voter gets to retain a copy of the encrypted vote which she can later confirm is correctly posted to a secure, append-only

© International Financial Cryptography Association 2016
J. Clark et al. (Eds.): FC 2016 Workshops, LNCS 9604, pp. 176–192, 2016.
DOI: 10.1007/978-3-662-53357-4_12

Web Bulletin Board (WBB). All the posted, encrypted ballots are then anonymously tabulated, either using mixes and decryption or exploiting homomorphic properties of the encryption to tabulate under encryption and then decrypt the result.

The assurance arguments are rather subtle though, and some people object to the use of crypto in voting on the grounds that the majority of the electorate will not really understand it and its role. Indeed, German Federal law, according to some interpretations, rules out the use of cryptography on the grounds that anyone should be able to understand the mechanisms without requiring any special knowledge. It is interesting therefore to explore the possibility of achieving some form of verifiability without the use of crypto. An early example of this is the article of Randell and Ryan [21] that uses scratch strips as an analogue of crypto. Another fine example is Rivest's ThreeBallot system [22].

Another approach is to have private ballot identifiers that allow voters to look up their vote in the clear on the WBB. Schneier in his book [27] for example suggests such an approach: voters are invited to invent their own random code and submit it with their vote. A slightly more sophisticated approach, in which the system and/or the voter's devices generates the numbers is presented in [1].

Introducing ballot identifiers has the appeal that it provides voters with a very simple, direct and easy-to-understand way to confirm that their vote is present and correct in the tally. There are however two significant drawbacks: care has to be taken to ensure that voters get distinct trackers and there is a danger of coercion. The first is an issue if, for example the system could identify two voters likely to vote the same way and assign them the same tracker. In this case it just posts one vote against this tracker and is free to stuff another vote of its own choice. The second danger is that a coercer requires the voter to hand over her tracker to allow him to check how she voted. Notice though that in this style of attack the coercer has a limited window of opportunity: he must request that the tracker be handed over before the results are published. It is this observation that we exploit to counter this threat: we arrange for the voters to learn their tracker numbers only after the vote/tracker number pairs have been posted to the WBB.

This paper presents a scheme that addresses both of these shortcomings by:

- Guaranteeing that voters get unique trackers.
- Arranging for voters to learn their tracker only after the votes and corresponding tracking numbers have been posted (in the clear).

We hope that by putting all the crypto under the bonnet, voters, election officials etc. may find such a scheme more acceptable that conventional E2E verifiable schemes that require voters to handle encrypted ballots. Here the voters just have to handle tracking numbers and votes in the clear. The scheme is also interesting in that it appears to shift the trust model for voter devices: in usual E2E schemes we need to worry about the voter's device encrypting the vote correctly. This typically necessitates complicating the protocol with *Benaloh challenges*, [3], or similar ballot assurance mechanisms. Now that voters get to check their vote in the clear, a misbehaving device can be detected more readily, resulting in a simpler voting ceremony.

A possible problem with the basic scheme, pointed out by Bill Roscoe, is that a coerced voter might by mis-chance choose the coercer's tracking number when she is deploying her coercion evasion strategy. Perhaps even more worrying is the possibility that the coercer will simply claim, falsely, that the tracker revealed by the voter is his and hence he "knows" that voter has not revealed her true tracker. This puts the voter in a very difficult situation. It seems that her best strategy is to stick to her guns and insist that she has revealed her true tracker. She does not know whether or not the coercer is telling the truth and indeed, ironically, the coercer does not have any means to prove to her that it is his tracker.

In large elections with a small number of candidates the odds of lighting on the coercer's tracker will typically be small (unless the coercer is backing a serious loser), but even the remote possibility may be worrying to some voters. If the coercer is not himself a voter the problem does not arise, but even here there may be an issue if many voters are being coerced. And, as remarked above, the coercer might claim, falsely, that the tracker is his.

It is not immediately obvious how to counter this danger, but in the full version [23] we present an enhancement to the basic scheme which counters this possibility, but where the tally is less transparent. An alternative version of the basic scheme which also counters the possibility of choosing the coercer's tracker is described in Sect. 8; however the cost is that coerced voters can no longer verify their cast vote.

The Selene scheme is in any case targeted at low coercion threat environments and so in such a context this problem could be regarded as minor. We suggest that, in some contexts, the benefits arising from the greater degree of transparency outweigh the rather remote threat. In any event, we show that the basic scheme still provides receipt-freeness.

It is worth noting that the constructions presented here could be thought of as a possible add-on to other schemes to provide a transparent form of verifiability. Indeed we could start with a simple, un-verifiable scheme that simply delivers (encrypted) votes to the server and render it verifiable by adding Selene constructs.

2 Background

Coercion can come in many flavours, from implicit, the coercer does not have to say anything, folk just know how they are expected to vote, to full-on: your personal coercer is on hand 24/7 to assist you in making the right voting choice. Making a voting system resistant to the latter form is extremely difficult, arguably impossible if the coercer really is observing the voter throughout the voting period. The Selene scheme is aimed at contexts where the coercion threat is closer to the former end of the spectrum: the coercer will issue some instructions and ask some awkward questions. Selene will mitigate such coercion attacks and at the same time allow the voters to directly verify that their vote is counted as intended.

3 Cryptographic Primitives

In this paper we will assume that the reader is familiar with signature schemes [13], threshold encryption [11], plaintext equivalence tests (PET), non-interactive zero-knowledge proofs of knowledge (NIZKPoK) [13] and verifiable shuffle protocols [20]. We further assume the existence of a secure Web Bullettin Board (WBB) [16]. We defer detailed descriptions of these to the the full version [23].

4 Related Work

E2E verifiable voting now has quite a long and rich literature, with many schemes having been proposed, both for in-person and remote, e.g. internet voting. Here we just mention some of the most closely related schemes. Note, Selene as presented here is intended for internet voting, but it would doubtless be straigforward to adapt it to in-person voting.

The most notable verifiable internet voting scheme is Adida's Helios, https://vote.heliosvoting.org/. Helios is not receipt-free, but recently the Belenios RF scheme, [8], has been proposed to provide receipt freeness.

Juels *et al.* [17] proposed a formal definition of coercion resistance and a credential-based mechanism to achieve this. The Civitas system, [7], http://www.cs.cornell.edu/projects/civitas/, implements this approach, with some enhancements.

The idea of voters having a private tracking number with which they can look up their vote in the clear on a bulletin board appears to go back the Schneier's "Applied Cryptography" book in which he suggests that voters choose a password to identify their vote. Much later the idea is revived for use in voting during ANR (Agence National de la Recherche) funding committee meetings. A scheme that has some similarities to Selene in that votes appear in the clear alongside identifying number, is Trivitas, [6]. Here, however, the clear-text votes appear on the bulletin board at an intermediate step, followed by further mixing and filtering. Hence the voters do not verify their vote directly in the tally. The goal is rather to allow voters to test the system by submitting dummy ballots.

5 The Set-Up Phase

The EA creates the threshold election key and keys share. Ideally this should be in a distributed, dealerless fashion [11]. We assume that any voter already has a PK/SK pair for an El Gamal encryption scheme and thus the PK of voter i has the form $\mathsf{pk}_i = g^{x_i}$, where x_i is her corresponding secret-key. When voters register for the election we assume that they, or more precisely their devices, create a fresh, ephemeral trapdoor key pair.

We now describe a distributed construction whose goal is to assign unique tracker numbers to the voters and inform them of their tracking numbers in a way that provides them with high confidence that it is correct but allowing them to deny it if coerced. We do this by generating trapdoor, Pedersen-style commitments to the tracking numbers. The tracking numbers could be rather sparse to be easily distinguishable, but can also be consecutive numbers $1, 2, \ldots$.

Distributed Generation of the Encrypted Tracker Numbers. The Election Authority publicly creates the tracking numbers n_i and computes g^{n_i} (to ensure that the resulting values fall in the appropriate subgroup) as well as the (trivial) ElGamal encryptions of the g^{n_i}: $\{g^{n_i}\}_{PK_T}$ and posts these terms to the WBB:

$$n_i, g^{n_i}, \{g^{n_i}\}_{\mathsf{pk}_T}$$

The (Mix) Tellers now put the last, encrypted terms through a sequence of verifiable, re-encryption mixes to yield:

$$\{g^{n_{\pi(i)}}\}'_{\mathsf{pk}_T}$$

These are now assigned to the voters' PKs

$$(\mathsf{pk}_i, \{g^{n_{\pi(i)}}\}'_{\mathsf{pk}_T})$$

Note that, thanks to the mixing, the assignment of these numbers to the voters is not known to any party, aside from a collusion of all the mix Tellers. Note also that as this is a verified mix, as long as all the input numbers are unique it is guaranteed that each voter will be assigned a unique (encrypted) number. We still need to ensure that the number revealed to each voter is the number assigned to them in the above construction; we will see this next.

5.1 Distributed Generation of the Tracker Number Commitments

Now, each Teller is required to produce n pairs of terms of the form:

$$(\{h_i^{r_{i,j}}\}_{\mathsf{pk}_T}, \{g^{r_{i,j}}\}_{\mathsf{pk}_T})$$

Here and in the following we have set $h_i := g^{x_i} = \mathsf{pk}_i$ for notational convenience.

We have to provide NIZKPoK proofs that these terms are well-formed, i.e. that the $r_{i,j}$ exponents in the two terms are indeed identical and known and that the Teller knows such value, we present these in the full version [23]. In addition we will have to assume that such proofs be non-malleable as we will explain later. Alternatively, one could also let the Tellers produce extra terms and perform a cut-and-choose audit.

Thus we now have a $n \times t$ array of such pairs, the columns corresponding to the Tellers and the rows to the voter ids. Now, for each voter, we form the product across the columns of the first elements to give:

$$\{h_i^{r_i}\}_{\mathsf{pk}_T} = \prod_{j=1}^{t} \{h_i^{r_{i,j}}\}_{\mathsf{pk}_T}$$

Where, due to the multiplicative homomorphic properties of ElGamal,

$$r_i := \sum_{j=1}^{t} r_{i,j}$$

Now we form the product of the $\{h_i^{r_i}\}_{\mathsf{pk}_T}$ and the $\{g^{n_{\pi(i)}}\}_{\mathsf{pk}_T}$:

$$\{h_i^{r_i} \cdot g^{n_{\pi(i)}}\}_{\mathsf{pk}_T} = \{h_i^{r_i}\}_{\mathsf{pk}_T} \cdot \{g^{n_{\pi(i)}}\}_{\mathsf{pk}_T}$$

This gives us the encryption under the Teller's PK of the trapdoor commitments to the tracking numbers: $(h_i^{r_i} \cdot g^{n_{\pi(i)}})$. We can now have a threshold set of Tellers perform verified, partial decryptions of these terms to reveal the commitments:

$$C_i := h_i^{r_i} \cdot g^{n_{\pi(i)}}$$

All of these steps are posted, along with NIZKPoK proofs and audits, to the WBB.

It seems that the Tellers cannot cheat in any effective way here aside from injecting invalid randoms which will result eventually in the voters being unable to open their commitment to a valid tracking number. But in any case, any such cheating will be detected by checks on the NIZKPoK proofs or random audits.

Now, for each voter there will be a tuple of terms posted to the WBB:

$$(\mathsf{pk}_i, \{g^{n_{\pi(i)}}\}_{\mathsf{pk}_T}, h_i^{r_i} \cdot g^{n_{\pi(i)}})$$

5.2 Voting

Voter V_i casts her vote in the form:

$$(\mathsf{Sign}_{\mathsf{V_i}}(\{\mathsf{Vote_i}\}_{\mathsf{pk_T}}), \mathsf{\Pi_i}),$$

where the ballot is signed either with the voter's true PK, or with her pseudo-PK if this has been configured (see the full version [23]), and Π_i is a non-interactive proof of knowledge of the plaintext. The signature is to avoid ballot stuffing, see e.g. [9]. The proofs of knowledge are needed to ensure *ballot independence* [10,12,29], by preventing an attacker copying, re-encrypting a previously cast vote as his own.[1] Note that in conjunction with Selene such a copying attack

[1] As Bernhard *et al.* [5] showed, it is possible to tweak the so called Enc+PoK paradigm (where one adds a proof of knowledge to an ElGamal ciphertext) to achieve non-malleable encryption that is sufficient for ballot independence. Another possibility is to resort to threshold Cramer and Shoup [28]. Note that any change will be completely transparent in Selene where the vote cast system can be essentially arbitrary.

would be particularly virulent: the attacker copies the victim's vote and casts it as his own. When the votes and trackers are revealed he sees exactly how the victim voted.

It is important that the server check for duplication of encrypted votes. It is also advisable to post the votes only once voting is closed. The signatures and proofs are checked for validity and, if valid, the encrypted votes are now paired off with the PK (and encrypted tracking number) with which they were signed. Double votes are handled according to the policy in operation, e.g. only the last vote cast by V_i is retained. Thus we get a list of tuples on the WBB:

$$(\mathsf{pk}_i, \{g^{n_{\pi(i)}}\}_{\mathsf{pk}_T}, (h_i^{r_i} \cdot g^{n_{\pi(i)}}), \mathsf{Sign}_{\mathsf{V_i}}(\{\mathsf{Vote_i}\}_{\mathsf{pk_T}}, \Pi_\mathsf{i}))$$

5.3 Mixing and Decryption

Now, for each row on the WBB, the second and fourth terms of these tuples are extracted and the signature and proofs striped off the fourth term. This gives pairs of the form:

$$(\{g^{n_{\pi(i)}}\}_{\mathsf{pk}_T}, \{\mathsf{Vote_i}\}_{\mathsf{pk_T}})$$

These are now put through a verifiable, parallel shuffle, e.g. [20]. Once this is done, a threshold set of the Tellers perform a verifiable decryption of these shuffled pairs. All of these steps along with the proofs are posted to the WBB. Thus, finally we have a list of pairs: tracking number, vote:

$$(g^{n_{\pi(i)}}, \mathsf{Vote_i})$$

from which the tracker/vote pair can immediately be derived: $(n_{\pi(i)}, \mathsf{Vote_i})$.

5.4 Notification of Tracker Numbers

For the notification of tracking numbers we will think of the Pedersen commitments whose construction we described earlier as forming the β component, i.e. the $h^r \cdot m$, of an ElGamal encryption under the voter's PK, but with the α component, i.e. the g^r, kept hidden. Thus we think of an ElGamal encryption as being represented:

$$(\alpha, \beta) := (g^r, h^r \cdot m)$$

The goal then is to reveal the α term to the voter in a deniable fashion.

Once the trackers and votes have been made available on the WBB for a sufficient period for the voters to note any alternative trackers as may be required to parry any attempted coercion, the Tellers send the voter V_j their share of the $g^{r_{j,i}}$ over a private channel:

$$T_j \rightarrow V_i : g^{r_{j,i}}$$

Once V_i's device has received these from all the Tellers it combines them to form g^{r_i}, the α term which along with the β term of the commitment $h_i^{r_i} \cdot g^{n_{\pi(i)}}$ to give the ElGamal encryption of $g^{n_{\pi(i)}}$ w.r.t. the voter's PK h_i:

$$(g^{r_i}, h_i^{r_i} \cdot g^{n_{\pi(i)}})$$

The voter can now decrypt this in the usual fashion using her secret key x_i, thus revealing $g^{n_{\pi(i)}}$ and hence $n_{\pi(i)}$.

The advantage of this construction is that it is unnecessary to authenticate the message notifying the voter of the α term. Authenticating these terms naively would introduce coercion threats. Designated Verifier Signatures or similar would be a way to sidestep such coercion threats, but they would significantly complicate the ceremony.

The point is that an adversary, even if colluding with all the Tellers, can only construct an α term that opens up to a valid tracker different from the true tracker of the voter with negligible probability. Stated formally:

Theorem 1. *If the* 1*-DHI assumption* [19] *holds, then there exists no PPT algorithm $\mathcal{A}$ which takes as inputs a description of a DH-group $\mathcal{G}$ along with a generator g for it, a set T of tracker number of polynomial size, two values $C = g^n h^r, h = g^x \in \mathcal{G}$ and outputs with non-negligible probability a term α such that C/α^x is of the form $g^{n'}$ where $n' \neq n$ is a valid tracker, $n' \in T$. Further, this holds true even if the algorithm is given n and r.*

A deeper discussion and a proof of the theorem can be found in the full version [23].

By contrast, the voter, or more precisely her device, with knowledge of the trapdoor, can compute an alternative $g^{r_i}{}'$ term that will decrypt to an alternative, valid tracker of her choice. Suppose that she wants her commitment to decrypt to the tracker value $m^* := g^{n^*}$, she inputs this to her device along with the commitment value β_i and the device computes the fake α term α':

$$\alpha' = \left(\frac{\beta_i}{m^*}\right)^{x_i^{-1}}$$

Note also that for the privacy of the tracking numbers we do not really need to encrypt the g^{r_i} terms as the trackers are still protected by the encryption under the voter's PK. However, it is still important to send these terms to the voter over a private channel to ensure that they are deniable.

Another potential attack lies in the fact that a Teller could create his g^{r_j} term with knowledge of the g^{r_i}'s terms of the other Tellers so that the product of all r_i's be known to him. This would be possible if the NIZKPoK proofs be malleable and in fact this is the case if care is not taken when applying the Fiat-Shamir heuristic. In the full version [23] we discuss how it is possible to use standard techniques to make a NIZKPoK non-malleable. We stress that by assuming that the NIZKPoK is non-malleable, the aforementioned attack is nullified.

6 The Voter Experience

A goal of the design of this protocol is to make the voter experience as simple and intuitive as possible. We assume that the voters already possess public (signing) keys and will create trapdoor keys during a registration phase. First we describe the ceremony in the case that the voter does not experience any coercion. Then we describe the steps needed to counter a coercer.

6.1 The Core Ceremony

– The voter receives an invitation to vote along with a ballot.
– The voter inputs her choice and her device encrypts this under the Election PK and signs this. The device sends this to the Election Server.

After a suitable period the tracking number/vote pairs are anonymised and decrypted and displayed on the WBB. The voters receive an invite to visit the WBB, but will only be necessary at this stage if the voter is being been coerced.

– After a suitable delay, the voter receives a notification of the α term, which she inputs to her device to allow it to extract her tracking number. Once she has this she can visit the WBB and confirm that her vote appears correctly against this tracker.

The last step is optional, to enable to voter to check that her vote was correctly recorded and entered into the tally. She can skip this if she is not interested in performing such a check.

6.2 The Ceremony in the Event of Coercion

If the voter is being coerced she needs to take some additional, coercion evasion steps, shown in italics:

– The voter receives an invitation to vote along with a ballot.
– The voter inputs her choice and her device encrypts this under the Election PK and signs this. The device sends this to the Election Server.
– *Once the (tracker, vote) pairs are displayed on the WBB she visits the WBB and notes down a tracking number that appears against the vote demanded by the coercer.*
– *The voter inputs this fake tracking number into her device and it outputs a fake α' term that coupled with her commitment, the β term of the ElGamal encryption of her tracker, will decrypt to the fake tracker.*
– After a suitable delay, the voter receives a notification of her “true” α term, which she inputs to her device to allow it to extract her tracking number from the commitment.
– *If the coercer demands that she reveal her tracking number she “reveals” the fake one. If he further demands that she reveals the alpha notification value, she reveals the fake α' she computed earlier.*
– Once she has her tracker she can visit the WBB and confirm that her vote appears correctly against this tracker.

Of course, she should also notify the appropriate authorities that coercion was attempted.

6.3 Selene as an Add-On

It is, however, interesting to note that the constructions described above could in many cases be added to an existing scheme, one without any verification features or perhaps one having conventional E2E verification involving encrypted receipts. Indeed, in some cases it could even be retro-fitted to an election that had already taken place. Suppose that a Helios vote had been conducted and contested. The trapdoor commitments to the trackers could be generated and associated to the voters as described above and the mixes and decryptions performed afresh. For this to work, the base scheme must use encryption such that we can run a parallel shuffle with the corresponding encrypted trackers.

In the full version [23] we discuess more enhancements to the basic scheme such as the use of re-encryptable signatures [8].

7 Analysis

In this section we give a brief, informal analysis of the security properties of Selene. A full, formal security analysis is postponed for future research.

7.1 Verifiability and Verification

If we think of Selene as an add-on to a base scheme, the universal verifiability of Selene is at least as strong as the base vote casting. In Sect. 5.2 this is a Helios like scheme, but as mentioned in Sect. 6.3 it could also be a more general scheme. Such schemes most often provide tallied-as-stored security, i.e. that the vote is tallied as cast by the device of the voter.

However, Selene could aso be added to a vote casting scheme without universal verifiability. Indeed, the strength of Selene is to provide additional individual direct verification that the vote is tallied as intended by the voter.

The security of the tracker construction relies on interested parties checking the proofs and calculations done on WBB as follows, but these are universally verifiable:

– Check that the trackers, n_i, written in plain on the WBB are indeed unique and their exponentiations g^{n_i} and the trivial encryptions thereof are correct (Sect. 5).
– Check the ZK proofs for the mix of the encrypted trackers (Sect. 5). This is to ensure both privacy and verifiability. We will elaborate on this in next subsection.
– Check the ZK proofs from the Tellers that the terms $\{h_i^{r_{i,j}}\}_{\mathsf{pk}_T}, \{g^{r_{i,j}}\}_{\mathsf{pk}_T}$ are well-formed. Further, it is checked that these are correctly multiplied together to give a commitment to the tracking number (Sect. 5.1). It can be shown (see the full version [23]) that an adversary with overwhelming probability cannot fake the α term, which the voter receives and uses together with the commitment to decrypt the tracker. This of course assumes that the voter's secret key $x_i = \log_g h_i$ is not known to the adversary. We will comment on this below.

– Check the proofs in the verifiable parallel shuffle of the voter/tracker pairs and their decryption (Sect. 5.3). As in a standard voting scheme using mixing for tallying this ensures that the tally is correct and in this case it further means that the tracker in the commitment is indeed the one shown next to the vote in the tally.

We conclude that if these checks are performed then a voter, who decrypts to a valid tracker, can be confident that this is the unique tracker assigned to her and the corresponding vote on the tally board is the vote stored encrypted on WBB.

More elaborate schemes also provide some security for the vote being stored as intended, even when the voter's device is malicious e.g. via Benaloh challenges [3] or by employing hardware tokens [15]. Selene, can however also provide verifiability in this respect. Checking the vote in the tally can reveal if a malicious device altered the intended vote. This requires that the voter checks her vote on an app or another device not controlled by the adversary. Further, the signature key used to cast the vote can also be different from the secret key x_i used to retrieve the tracker. In this case the device used to cast the vote does not even need to know x_i. This means that the adversary cannot calculate an alternative value for the α term and it will be more difficult to launch an attack. A voter can then even use the same device to receive the α term, then store it and then reveal the secret key to get the tracker. Later the voter can then check if it gives the same tracker on another device.

7.2 Ballot Privacy

The Selene scheme requires that the underlying ballot casting mechanism provides good privacy. Thus the encryption algorithm and its implementation used to encrypt the vote should ensure the secrecy of the vote. The first mix of the encrypted trackers means that only an adversary controlling all the mix servers would know the association of the tracking numbers to the voters, assuming that the proofs of the mixing have been checked. The posted commitments to the tracking numbers are perfectly hiding unless the adversary colludes with all the Tellers. Finally the parallel mix preserve ballot privacy for both the vote and the tracker just like in a standard vote scheme using tallying via mix nets. Finally, the α term, if this should come into the possession of an adversary, does not reveal the tracker since it just a part of an ElGamal encryption of the tracker.

7.3 Receipt-Freeness

In their seminal paper Benaloh and Tuinstra [4] defines receipt-free (which they call uncoercibility) informally as "no voter should be able to convince any other participant of the value of its vote".

If the vote casting scheme is receipt-free, e.g. by employing the model of BeleniosRF [8] for the vote casting, then Selene is receipt-free. Basically the extra information that the voter has in Selene is the unique tracking number. However, the voter can simply fake this (and importantly the corresponding α term) since the tally board is presented before the tracker retrieval. We do need to assume that he attacker cannot monitor the communication of the α terms to the voters. As mentioned before, it can happen that the voter chooses a fake tracker which coincide the tracker of the coercer, however, this does not constitute a proof of how she voted, it just undermines her claim to that tracker and associated vote.

To which extent this makes Selene vote buyer resistant is a subject of future research. The point is that even though the voter cannot prove her vote, she does have extra information, namely the tracker which is unique to her.

We also mention that Italian style (aka signature) attacks may be possible here when we are dealing with complex ballots. For some voting methods we may be able to counter this by splitting up the ballot into components and mixing separately.

7.4 Coercion: Threats and Mitigation

For Selene to be coercion resistant, we firstly need that this is true for the vote casting part. Some degree of coercion resistance can be obtained by combining BeleniosRF [8] with vote updating. Another possibility for partial coercion resistance is to use the scheme by Kulyk et al. [18] where each voter can cast several vote values and only the sum of these will count in the end. The total number of votes are hidden in a cloud of null votes which any participant can cast for the voter.

For Selene the extra tracker verification step however also opens up a coercion possibility: the coercer can demand to observe the receipt of the $g^{r_{j,i}}$. Of course the voter can always create a fake term $g^{r'_{j,i}}$ and pretend to the coercer that this is the term that was sent to her, see Sect. 5.4. Further, the terms are sent at randomized times and the coercer will thus have to intensively follow the voter. However, the possibility of receiving a wrong term while the coercer is present, might be discouraging for the voter. A possibility to circumvent this is to allow voters to secretly contact the voting authorities to request that only the fake $g^{r_{j,i}}$ term that the voter has calculated be communicated back to her. They are now safe from the coercion threat, but a coerced voter have lost the individual verifiability. This suggests a novel form of coercion resistance, distinct from the conventional one in which the voter gets to cast her intended vote and to verify it, or *coercion evidence*, [14], in which she gets to verify her vote but it might be nullified. Here she gets to cast her intended vote but if coerced may lose the possibility to verify it.

The coercion problem might escalate if the coercer is colluding (or pretends to be) with one of the Tellers. The voter then has to guess which $g^{r_{j,i}}$ to fake (this is incidentally also a problem in Civitas [7]). In the BeleniosRF construction there is a voting authority which is trusted for the receipt-freeness, and in this case we can circumvent this danger by letting this authority receive the $g^{r_{j,i}}$ terms and only forward the g^{r_i} to the voter.

True coercion-resistant vote schemes often work with credentials, e.g. Civitas [7]. The voter knows the true credential and can provide the coercer(s) with fake credential(s). Where Civitas is not directly compatible with Selene, one can imagine to combine its credential construction and the extra null votes of [18] to create a true coercion-resistant scheme compatible with the tracker construction. In this case the extra credentials can also be used to make the tracker retrieval coercion-resistant. A scheme could be as follows. After the tally board is created we allow a certain time for the voters to note the trackers, construct fake α-terms and contact the voting authorities privately with these terms. After this time the voter can log in to the voting system to get the α term, however the credential is also used in this process. The voting authority provide the true α term if the correct credential is used. If a fake credential is used, the system outputs the corresponding faked α-term which has been provided by the voter.

7.5 Dispute Resolution

Dispute resolution, the ability to determine the cheating or malfunctioning component or party when an error is reported, is quite hard to achieve, especially in the internet voting context. In Selene this appears to be tricky. If a voter claims that the vote corresponding to their tracker is not what they cast, it is hard to determine if it is the voter who is lying or mis-remembering, or the device or the system that cheated. But this is a problem with the tracking number approach anyway.

If a voter insists that the vote on the WBB is wrong, we could resolve this if the voter is prepared to sacrifice their ballot privacy by allowing threshold decryptions of their ballot for example. This has to be performed with great care and suitable controls, and presumably *in camera* to avoid introducing coercion opportunities. The use of voting codes may help here, but this necessitates mechanisms to distribute these to the voters in a secure fashion and complicates the scheme.

8 Alternative Selene Scheme

We will now briefly describe an alternative version of the scheme which dispels the chance of being caught lying about a faked tracker, but where the coerced voters loose their ability to verify their vote. The idea is that the voting authority adds $f \cdot c$ extra fake trackers, where f is a number greater than the expected number of coerced voters. These trackers are added in the clear before the mixing of the trackers and we thus in total have $v + f \cdot c$ trackers. All the trackers are sent through a first mixing giving the anonymised encrypted tracking numbers on the BB

$$\{g^{n_{\pi(a)}}\}'_{PK_T}\ , \quad a = 1, \ldots, v + f \cdot c$$

The first v trackers are used as in the basic Selene construction. The remaining $f \cdot c$ trackers are collected into f sets $\{g^{n'_{s,k}}\}'_{PK_T}$ where $s = 1, \ldots, f$ and

$k = 1, \ldots, c$. For each set, the c trackers are on the BB assigned to a vote for each candidate in a public fashion using trivial encryptions

$$(\{g^{n'_{s,1}}\}_{PK_T}, \{\mathsf{Cand}_1\}_{\mathsf{PK_T}}), \ldots, (\{\mathsf{g}^{\mathsf{n}'_{\mathsf{s,c}}}\}_{\mathsf{PK_T}}, \{\mathsf{Cand}_\mathsf{c}\}_{\mathsf{PK_T}}).$$

In the final construction of the tally board on BB, the extra trackers are added to the ones which have gone through the basic Selene construction and are mixed along with these. This means that the resulting tally board contains f extra votes for each candidate corresponding to the $f \cdot c$ fake trackers. Due to the first and final mixing nobody at this stage knows which trackers are the fake ones.

After revealing the trackers, coerced voters can now contact the voting authority via an anonymous channel. The voting authority will then request a fake set of c trackers to be jointly decrypted by the Tellers, and it will send these trackers to the voter. Further it will instruct the Tellers not to inform the corresponding voter of the real α term. It will use a unique fake set for each coerced voter. The coerced voter can now use the unique tracker of choice to show to the coercer, and she can also compute the corresponding fake α term. The voter gets c trackers to sidestep an anonymity issue: if a voter asks for a fake tracker for a specific candidate, she probably did not vote for that candidate.

The coerced voter cannot get her real tracking number. The reason is that the coercer would then demand to see two unique tracking numbers for the candidate of his choice. This means that we have a new type of weak coercion-resistance where the un-coerced voters can verify, but coerced voters can cast the vote of their choice, but loose the ability to directly verify this vote. In the construction above each Teller is trusted for coercion-resistance, however, with a bit more elaborate construction this trust could be moved to the voting authority alone.

9 Conclusions

We present a new voting protocol, based on the idea of tracking numbers but with the twist that voters do not learn their number until after voting has finished and the tracker/vote pairs have been posted to the bulletin board. This counters the usual coercer attack on such tracking number systems: the coercer demands that the voter hand over her tracking number before the results are posted. We also provide a mix net construction that ensures that each voter gets a unique tracking number, preventing the attack of assigning the same tracker to voters likely to vote the same way. The construction ensures a high level of assurance that the voter receives the correct tracker while ensuring that this is deniable to a third party.

The resulting scheme provides a good level of verifiability and coercion resistance while at the same time providing a very direct and simple to understand mechanism for voter verification. The protocol is not crypto free, but the crypto is kept under the bonnet for ordinary voters, and in particular the voter verification step involves just tracking numbers and votes in the clear. Voters do not have to handle encrypted ballots as is the case for previous E2E verifiable schemes. A further advantage appears to be that we avoid the need to audit the ballots created by the voter's device. Typically this necessitates the introduction of some kind of cut-and-chose protocol into the voting ceremony, significantly complicating the voter experience. Now, because the voter gets to check her vote in the clear we can sidestep this complication, but at the cost of incurring dispute resolution issues.

For future research, it would be interesting to perform a usability experiment on the Selene protocol to gauge the user experience compared to other e-voting schemes. We also plan to investigate mechanisms to provide cleaner dispute resolution.

In is interesting to note that the Selene construction can be thought of as an add-on to an existing non-verifiable scheme, or indeed a conventional E2E verifiable scheme for which people want a greater degree of transparency in the verification. Indeed Selene could even be retrofitted to a cryptographic election that has been contested. Note further that an option is to run the basic Selene scheme, but if a significant level of coercion is reported before and during the vote casting period, the Selene II constructions (presented in the full version of the paper) could be dynamically added to the WBB give the higher degree of coercion resistance.

Acknowledgements. We would like to thank Sunoo Park, Bill Roscoe, Mark Ryan and Richard Stallman for interesting discussions and suggestions. Further, Vincenzo Iovino is supported by the National Research Fund, Luxembourg, and Peter B. Rønne is supported by the ANR/FNR project Sequoia ANR-14-CE28-0030-01.

References

1. Arnaud, M., Cortier, V., Wiedling, C.: Analysis of an electronic boardroom voting system. In: Heather, J., Schneider, S., Teague, V. (eds.) Vote-ID 2013. LNCS, vol. 7985, pp. 109–126. Springer, Heidelberg (2013)
2. Ben-Nun, J., Fahri, N., Llewellyn, M., Riva, B., Rosen, A., Ta-Shma, A., Wikström, D.: A new implementation of a dual (paper and cryptographic) voting system. In: 5th International Conference on Electronic Votin (eVOTE) (2012)
3. Benaloh, J.: Simple verifiable elections. In: Wallach, D.S., Rivest, R.L. (eds.) USENIX/ACCURATE Electronic Voting Technology Workshop, EVT 2006, Vancouver, BC, Canada, 1 August 2006. USENIX Association (2006)
4. Benaloh, J.C., Tuinstra, D.: Receipt-free secret-ballot elections (extended abstract). In: Leighton, F.T., Goodrich, M.T. (eds.) Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, 23–25 May 1994, Montréal, Québec, Canada, pp. 544–553. ACM (1994)
5. Bernhard, D., Pereira, O., Warinschi, B.: How not to prove yourself: pitfalls of the Fiat-Shamir heuristic and applications to Helios. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 626–643. Springer, Heidelberg (2012)
6. Bursuc, S., Grewal, G.S., Ryan, M.D.: Trivitas: voters directly verifying votes. In: Kiayias, A., Lipmaa, H. (eds.) VoteID 2011. LNCS, vol. 7187, pp. 190–207. Springer, Heidelberg (2012)
7. Clarkson, M.R., Chong, S., Myers, A.C.: Civitas: a secure voting system. In: IEEE Symposium on Security and Privacy (2008)

8. Cortier, V., Fuchsbauer, G., Galindo, D.: Beleniosrf: a strongly receipt-free electronic voting scheme. IACR Cryptology ePrint Archive 2015:629 (2015)
9. Cortier, V., Galindo, D., Glondu, S., Izabachène, M.: Election verifiability for Helios under weaker trust assumptions. In: Kutyłowski, M., Vaidya, J. (eds.) ICAIS 2014, Part II. LNCS, vol. 8713, pp. 327–344. Springer, Heidelberg (2014)
10. Cortier, V., Smyth, B.: Attacking and fixing Helios: an analysis of ballot secrecy. In: Proceedings of the 24th IEEE Computer Security Foundations Symposium, CSF 2011, Cernay-la-Ville, France, pp. 297–311, 27–29 June 2011
11. Cramer, R., Gennaro, R., Schoenmakers, B.: A secure and optimally efficient multi-authority election scheme. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 103–118. Springer, Heidelberg (1997)
12. Gennaro, R.: Achieving independence efficiently and securely. In: Anderson, J.H. (ed.) 14th ACM Symposium Annual on Principles of Distributed Computing, pp. 130–136. Association for Computing Machinery, August 1995
13. Goldreich, O.: Foundations of Cryptography: Basic Applications, vol. 2. Cambridge University Press, Cambridge (2004)
14. Grewal, G.S., Ryan, M.D., Bursuc, S., Ryan, P.Y.A.: Caveat coercitor: coercion-evidence in electronic voting. In: 2013 IEEE Symposium on Security and Privacy, SP 2013, Berkeley, CA, USA, 19–22 May 2013, pp. 367–381. IEEE Computer Society (2013)
15. Grewal, G.S., Ryan, M.D., Chen, L., Clarkson, M.R.: Du-vote: remote electronic voting with untrusted computers. In: Fournet, C., Hicks, M.W., Viganò, L. (eds.) IEEE 28th Computer Security Foundations Symposium, CSF 2015, Verona, Italy, 13–17 July 2015, pp. 155–169. IEEE (2015)
16. Heather, J., Lundin, D.: The append-only web bulletin board. In: Degano, P., Guttman, J., Martinelli, F. (eds.) FAST 2008. LNCS, vol. 5491, pp. 242–256. Springer, Heidelberg (2009)
17. Juels, A., Catalano, D., Jakobsson, M.: Coercion-resistant electronic elections. In: Proceedings of the 2005 ACM Workshop on Privacy in the Electronic Society, WPES 2005, Alexandria, VA, USA, pp. 61–70, 7 November 2005
18. Kulyk, O., Teague, V., Volkamer, M.: Extending Helios towards private eligibility verifiability. In: Haenni, R., Koenig, R.E., Wikström, D. (eds.) VoteID 2015. LNCS, vol. 9269, pp. 57–73. Springer, Heidelberg (2015)
19. Pfitzmann, B., Sadeghi, A.-R.: Anonymous fingerprinting with direct non-repudiation. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 401–414. Springer, Heidelberg (2000)
20. Ramchen, K., Teague, V.: Parallel shuffling and its application to prêt à voter. In: 2010 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections, EVT/WOTE 2010, Washington, D.C., USA, 9–10 August 2010
21. Randell, B., Ryan, P.Y.A.: Voting technologies and trust. In: IEEE Symposium on Security and Privacy, pp. 50–56 (2006)
22. Rivest, R.L.: The ThreeBallot Voting System. https://people.csail.mit.edu/rivest/Rivest-TheThreeBallotVotingSystem.pdf
23. Ryan, P.Y.A., Rønne, P.B., Iovino, V.: Selene: voting with transparent verifiability and coercion-mitigation. IACR Cryptology ePrint Archive, 2015:1105 (2015)
24. Ryan, P.Y.A., Teague, V.: Pretty good democracy. In: Workshop on Security Protocols (2009)
25. Ryan, P.Y.A., Schneider, S.A.: Prêt à voter with re-encryption mixes. Technical report CS-TR-956, University of Newcastle (2006)
26. Scantegrity Team. Scantegrity. http://www.scantegrity.org/papers/whitepaper.pdf

27. Schneier, B.: Applied Cryptography - Protocols, Algorithms, and Source Code in C, 2nd edn. Wiley, Hoboken (1996)
28. Shoup, V., Gennaro, R.: Securing threshold cryptosystems against chosen ciphertext attack. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 1–16. Springer, Heidelberg (1998)
29. Wikström, D.: Simplified submission of inputs to protocols. In: Ostrovsky, R., De Prisco, R., Visconti, I. (eds.) SCN 2008. LNCS, vol. 5229, pp. 293–308. Springer, Heidelberg (2008)