Securing the Foundations of Democracy

Peter Y.A. Ryan(✉)

University of Luxembourg, Luxembourg City, Luxembourg
peter.ryan@uni.lu

Abstract. Recent events have highlighted numerous threats to democracy, in particular the 2016 US presidential election is mired in controversy. Allegations of Russian interference with the campaigns, in particular hacking and selective leaking of emails from the Democratic campaign management, possible hacking of electronic voting and tabulating. Alongside this we have challenges to democratic debate due to "fake news", information bubbles, the chilling effect of mass surveillance etc. All of this suggests that we need to have a major rethink of how democracy should function effectively in the digital age.

In a short article we cannot hope to address all of these threats, but rather we focus on just one aspect, arguably the keystone of democracy: making secure the conduct of elections. In particular we outline approaches to making elections verifiable and accountable, while guaranteeing ballot privacy and coercion resistance.

1 Introduction

The election of Donald Trump in 2016 to the most powerful office on the planet brought into sharp relief the strains that digital technologies are placing on the democratic process. In theory such technologies could enrich the democratic process by for example facilitating the dissemination of information and fostering debate. In practice we have seen that such technologies open up new and poorly understood threats. In the case of the US presidential election we have witnessed hacking of the email servers of the Democratic campaign committee, of registers of voters and apparent attempts to hack voting and tabulation machines.

Besides all these threats to the collecting, recording and counting of votes, we are seeing a raft of threats to the surrounding processes supporting the dissemination of news and information as well as informed debate. We hear of the prevalence of *fake news* and the rise of news disseminated via social media, resulting in *information bubbles* that serve to reinforce prejudices and preconceptions. All of this undermines the informed debate and decision making that should be the bedrock of a healthy democracy.

Clearly, in a short, rather technical, paper we are not going to able to address all of these issues, rather we just attempt to overview some approaches that have emerged from the information assurance community to address the security of the election process. In particular I will give a high-level overview of approaches

P. Farshim and E. Simion (Eds.): SecITC 2017, LNCS 10543, pp. 52–66, 2017.
https://doi.org/10.1007/978-3-319-69284-5_5

that go under the heading of *end-to-end verifiable* (E2E V) or sometimes *fully auditable* schemes. Here to goal is to provide strong guarantees that all legitimately cast votes are accurately counted while preserving ballot privacy, receipt freeness and, ideally, coercion resistance, while keeping trust assumptions to a minimum. After giving an outline of the design principles behind E2E V schemes, I will go into more detail on a new scheme, called Selene, that seeks to make the voter verification much more transparent and intuitive, while maintaining coercion mitigation.

1.1 Trust Assumptions in Conventional Voting Systems

Conventional voting systems typically require a significant level of trust to be placed in various components, which may be technological or human. Old fashioned voting with paper ballots and hand-counting involve trust in the people and procedures handling the ballot boxes, doing the counting etc. This can be partly offset by allowing independent observers but these will not be infallible and some level of trust still needs to be placed in their competence and honesty. Nonetheless, it can usually be argued that pulling off large-scale, fraud in a way that is likely to be undetected is extremely hard.

In the case of DREs however it is clear to anyone with even a modest understanding of the fragility of software and network security that large-scale, virtually undetectable fraud is quite easy to pull off, by simply tweaking a few lines of code. Indeed at the time of writing the DefCon conference staged demonstrations of hacking of various US voting machines, and showed that in some cases hackers could take over a machine in only minutes.

The response of most security experts to the use of voting computers is either, as was the case in the Netherlands, to demand that they be banned, or, in the US, to insist that any DRE must be supplemented by a Voter Verified Paper Audit Trail (VVPAT). This is essentially a printer at the side that prints the voters choice to paper, that can be checked by the voter and confirmed if correct. As long as the resulting paper audit trail is well curated, this creates a record that can be used for example for Risk Limiting Audits, [17]. If a link is maintained between each paper ballots and its digital representation used in the electronic tally then highly efficient *comparison audits* are possible: a typically very small, random sample of the ballots can be taken and if for all ballots the paper and electronic representations agree then a high level of confidence in the declared result can be obtained.

Clearly, if we are prepared to completely trust an authority to handle the votes correctly and ensure ballot privacy then we trivially solve the problem. However, nobody should be comfortable placing such trust for such a critical service, and for cryptographers and security specialists having to place such heavy trust is an anathema. The goal then is to try to make the processing of the votes as transparent as possible while respecting the privacy of votes. Steering a course between these conflicting requirements with minimal trust assumptions is what makes this an immensely challenging topic. Add to this the requirement that the system should be extremely easy to use and understand by every member of the electorate, who might use the system once every few years, and the fact that voters may cooperate with coercers or vote-buyers and you have arguably the biggest challenge facing information security engineers.

And we haven't even got onto the challenges of internet voting yet, with all the inherent insecurity of the internet, dangers arising from corrupted client devices, and the impossibility of preventing coercers observing the voters.

What cryptographers have sought to do, in numerous proposed schemes, is to use the rich toolkit of modern cryptography to make the execution of the election as transparent as possible. Such schemes seek to ensure that any errors or corruption in the handling of votes are detectable in a way that is, as far as possible, observable and verifiable by all. The difficulty with observing a conventional voting system is that you are monitoring an ephemeral process, if you miss some sleight of hand then the evidence is gone. In other words, in the terminology of Stark and Wagner, [18], we seek to make elections *evidence-based*: as the election unfolds, the system and authorities are required to generate sufficient evidence, of sufficient quality, to convince any reasonable sceptic, above all, the losers of the correctness of the announced result.

2 End-to-End Verifiable Schemes

In this section I give a very high-level indication of the key ideas behind the E2E V approach. The goal of such schemes is to enable each voter to be able to confirm to their own satisfaction that the vote that they cast is accurately included in the final tally. Immediately the astute reader will see that this is going to be tricky: if we provide ways to convince the voter that their vote is counted how are we going to avoid this being used to convince a coercer or vote-buyer? This is indeed immensely delicate, and this is where the magic of modern crypto comes to our aid.

The key idea is that when the vote is cast, a form of receipt is generated that carries a suitably encrypted or encoded representation of the vote. The voter gets to keep a copy of this receipt, or *protected ballot* as Rivest has suggested it be called. Copies all such receipts are posted to a *Public Bulletin Board*, (PBB). The PBB needs some explanation: it should have the following properties:

- it should be visible to all, and all should guarantee a consistent view of its contents to all,
- anything posted to the WBB should be guaranteed to remain posted and unaltered, i.e. it should be append-only,
- only appropriate authorities should be able to post items.

The voter gets to retain a copy of the encrypted vote which she can later confirm is correctly posted to the 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 point of encrypting the vote is of course to ensure that even of the voter shows it to someone else the privacy of the vote remains.

A number of E2E V schemes have been proposed and some even implemented and deployed, for example, prominent in-person schemes include Prêt à Voter [15] Wombat [2] and Scantegrity II [16], StarVote [8], while internet schemes include Helios https://vote.heliosvoting.org/, Civitas [5] and Pretty Good Democracy [14].

2.1 Verifiable Tabulation

Once we have an agreed set of encrypted votes, the extraction of the tally in a veritable fashion while ensuring ballot secrecy involves subtle but well-understood cryptographic techniques. For example, the set of encrypted votes can be put through a number of verifiable mixes to ensure anonymity and then verifiably decrypted. The result is the set of decrypted votes as cast, which can be counted up by anyone, but with any link back to the original receipts obliterated. An alternative approach is to exploit homomorphic properties possessed by many probabilistic encryption algorithms: exponential ElGamal and Paillier enjoy additive homomorphic properties: the product of encrypted plaintexts equals the encryption of the sum of the plaintexts:

$$\prod_i \{x_i\}_{PK} = \{\sum_i x_i\}_{PK}$$

This is very handy property for voting systems as it allows us to sum up votes under the wraps of encryption and then just decrypt the final counts in one shot. No individual encrypted ballots are revealed so ensuring ballot secrecy. To take a simple example, suppose that we are running a referendum, we encode a *yes* vote as a $+1$ and a *no* vote as a -1. To compute the result we take the product of the encrypted votes and then decrypt the result, if this is positive that the yeas have it, if negative then the nays carry the day. More complex elections, for example, with multiple candidates, can similarly be conducted with suitable encodings of the votes.

2.2 Ballot Auditing

The really interesting, and challenging part of designing an E2E V voting system is in the creation of the encryptions of the votes. It is essential that the voter be confident that his or her vote is correctly encrypted in the receipt, and this assurance must be provided in a way that cannot be conveyed to another party. It is far from obvious how a voter is to be sure that random string of symbols printed on the receipt is a correct encoding of the intended vote. Most schemes try to tackle this with some form of *cut and choose* protocol: the voting machine commits, in print say, to a number of encryptions, k say, of the vote, and the voter gets to chose to audit $k-1$ of these. If all the audited encryptions turn out to hold the correct vote then this provides assurance that remaining one is also correct, and this can now be cast with some confidence.

An alternative approach, referred to as *Benaloh challenges*, [3], is to do something similar but in a sequential fashion: the device generates an encryption of the vote and the voter is now given a choice as to whether to audit or cast this ballot. If she audits, and is happy with the result, she obtains a fresh encryption and again has the choice: audit or cast. This can go on in principle indefinitely until she is sufficiently confident the encryption device is behaving correctly, at which point she casts the vote and the process is complete.

An obvious question is: why not just audit one encryption, and cast this if it correct? The answer to this is that auditing typically produces a proof of the plaintext, which could then be presented to a coercer or vote-buyer. A highly ingenious scheme, MarkPledge [1], does allow the cast ballot to be audited, but this involves this voter participating in an interactive zero-knowledge proof protocol with the device in the booth. The real, interactive proof transcript for the voter's choice, generated with the voter, is masked by fake proof transcripts for the other candidates. Only the voter, who participated in the creation of the receipt in the booth, knows which was the real interactive proof. The scheme is technically quite brilliant, but the resulting complexity and lack of usability prevented wide-scale uptake.

In Prêt à Voter, printed ballot forms are generated with the candidates listed in a randomised order. Each ballot has an independently generated order and carried an encrypted representation of the order. In the privacy of the booth, the voter extracts the form from a tamper proof envelope and applies the appropriate marks against the candidate(s) of choice, an X or ranking etc. The plaintext list of the candidates is detached and destroyed. The result is an receipt carrying the vote in encrypted form: without knowledge of the order in which the candidates were presented in this particular ballot the vector of marks cannot be interpreted.

When presented with a ballot form, sealed in an envelope, the voter can opt either to cast her vote using the ballot or to audit the ballot. If she adopts to audit, and is happy with the outcome, she takes another form and again has the choice, Benaloh style. Later, during tabulation a threshold set of Tellers cooperate to extract the vote, taking appropriate care to protect the privacy of the vote.

This approach has some appealing features, notably:

– the vote is not communicated to any device, so sidestepping side-channel threats.
– ballot auditing is very clean and privacy preserving: you simply audit the blank form, if it is well-formed in the sense that the plaintext printed order agrees with the encrypted order then a vote cast with this form would be correctly encrypted.

The second point means that Prêt à Voter ballot auditing provides strong dispute resolution: there is no question of whether the fault lies with the voter, the ballot for is either well-formed or it is not, and this is wholly independent of the voter and the vote. Furthermore, this means that ballot audits can be performed by anyone: we can have independent auditors and observers performing random audits on forms, in addition to the audit performed by the voters.

3 Public Acceptance of E2E V Schemes

The general approach sketched above has a lot of technical merit and offers high assurance of accuracy of the tally along with guarantees of ballot privacy and coercion resistance with minimal trust assumptions. 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. People are often troubled at not being able to identify their vote in the clear in the tally, and seem unconvinced when it is pointed out that if this capability were to be provided it would open the system up to coercion and vote buying.

Ironically, the fact that errors or corruption are made detectable often does not seem to inspire trust. A good scheme should be both trustworthy and trusted. All too often we see commercial schemes that are not trustworthy apparently trusted and conversely highly trustworthy schemes which fail to inspire trust. While developing and trialling Prêt à Voter colleagues at the University of Surrey conducted focus groups. The groups were given a description of the scheme and its security guarantees and were asked what they thought of it. Many answered along the lines of: "it is all very well offering a scheme that can detect when things go wrong, but surely it would be better to design one that cannot go wrong".

All of these considerations suggest that it is interesting 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 [11] that uses scratch strips as an analogue of crypto. Another fine example is Rivest's ThreeBallot system [12].

Another objection often raised against such schemes is the point that verifiable does not automatically mean verified. For an election to be deemed verified we must be able to show that voters and observers did indeed perform the checks in sufficient numbers and with sufficient diligence. It is essential therefore that the various checks be as easy to perform as possible and well motivated. Furthermore we need reliable ways to monitor the levels of checking. A question that may spring to mind at this point is: why not just automate the checks? The answer to this is that if we automate them then we are thrown back into having to place trust in the processes performing the checks. Our goal here is to make the electorate themselves the bedrock on which the trust is based.

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 will just mention some of the most closely related schemes.

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, [6], has been proposed to provide receipt freeness.

Juels *et al.* [9] proposed a formal definition of coercion resistance and a credential-based mechanism to achieve this. The Civitas system, [5], 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, [4]. 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 final tally.

5 Selene

In this section I provide an overview of a new scheme that aims to provide voter verifiability but in a much more intuitive way, and which avoids voters handling encrypted receipts. A full description can be found at [13]. The scheme is based on an old and simple idea: voters have a private tracker number which allows them to identify their vote in the tally on the PBB. Earlier we remarked that this poses obvious problems in terms of coercion and vote buying, however the Selene scheme introduces some new twists that at least mitigates these issues.

Such an approach provides voters with a very simple, direct and easy-to-understand way to confirm that their vote is present and correct in the tally, but we must ensure that voters get distinct trackers and, as remarked above, there is a danger of coercion and vote buying. 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 insert in the tally 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 must request that the tracker be handed over before the results are published. If he asks after the trackers and votes have been published the voter has the opportunity to pick an alternative tracker pointing to the coercer's vote and claim it as her own. It is this observation that we exploit to counter this threat: we arrange for the voters to learn their tracker numbers only after the information has been posted to the WBB. The Selene scheme addresses both of these shortcomings: by guaranteeing that voters get unique trackers and arranging for voters to learn their tracker only at some time after the votes and corresponding tracking numbers have been posted (in the clear).

The hope is that by putting the crypto under the bonnet, voters, election officials etc. may find such a scheme more acceptable that conventional E2E verifiable schemes. 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. As observed above, this typically necessitates complicating the voting experience with Benaloh challenges, or similar ballot assurance mechanisms. Now voters get to check their vote in the clear, a misbehaving device can be detected more readily, resulting in a simpler voting ceremony. The downside of this is that, in the event that a voter contests the posted vote, it is harder to determine which is at fault, the system or the voter. This issue, usually referred to as *dispute resolution* is a further desirable property of a good voting system, and we will return to this later.

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 awkward situation.

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. However, the other scenario: the coercer claiming, falsely, that the tracker is his, places the voter in a difficult situation. Note that this can only arise of the coercer is himself a voter.

It is not immediately obvious how to counter this danger, but an enhancement to the basic scheme which counters this possibility is described in [13]; however it comes at a cost of a less transparent tally.

The Selene scheme is in any case targeted at low coercion threat environments. We argue that, in such contexts, the benefits arising from the greater degree of transparency outweigh a rather remote and mild threat. In any event, it can be shown that the scheme provides receipt-freeness.

5.1 Selene as an Add-On

It is interesting to observe that the Selene constructions could in many cases be added to an existing voting 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. Indeed, in our presentation below we will abstract away from details of exactly how votes are cast, validated etc.

6 The Set-up Phase

The EA creates the threshold election key and keys share. Ideally this should be in a distributed, dealerless fashion [7]. When voters register for the election we assume that they, or more precisely their devices, create a fresh, ephemeral trapdoor key pair. For the purposes of this paper we will leave aside the question of how voters and ballots are authenticated. It might be for example that voters all have signing keys, or they are provided with some form of credential.

We now describe the construction whose goal is to:

- ensure that each voter has a unique tracker committed to them,
- and inform voters 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.

6.1 Distributed Secret Assignment of Tracker Numbers

The tracking numbers could be short strings of digits, but could also be consecutive numbers $1, 2, \ldots$. The Election Authority (EA) publicly creates a list of the tracking numbers n_i and posts this to the PBB. Everyone can confirm that the elements of the list are pairwise distinct. EA now computes g^{n_i} (to ensure that the resulting values fall in the appropriate subgroup) and the ElGamal encryptions under the Teller's threshold public key PK_T of the g^{n_i}: $\{g^{n_i}\}_{PK_T}$ and posts these terms to the WBB:

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

These initial encryptions could be trivial, i.e. using a known randomisation such as $r = 1$ to allow universal verifiability of this step. Mix Tellers now put the encrypted terms through a sequence of verifiable, re-encryption mixes to yield:

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

where π denotes the permutation that results from the sequence of permutations applied by the mix tellers, and $\{X\}'$ denotes re-encryption of $\{X\}$. These are now assigned to the voters' IDs (or perhaps pseudo-IDs):

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

Thanks to the multiple shuffles, the assignment of these numbers to the voters is not known to any party, only a collusion of all the mix Tellers could determine the assignment. Note also that as these are verified mixes, as long as all the input numbers are unique the assigned (encrypted) numbers will be unique to each voter.

6.2 Generation of the Tracker Number Commitments

We now need to generate, for each voter, the trapdoor commitments to the tracker. [13] gives a rather elaborate, distributed construction, but here we give a simpler construction based on a suggestion from D Wikström that uses calls to general purpose, verifiable mix net, such as Verificatum, see http://www.verificatum.com/. Using the parallel mixing facility we can generate for voter V_i a pair of ElGamal ciphertexts:

$$(u_i, v_i) = (\{g^{r_i}\}_{PK_T}, \{h_i^{r_i}\}_{PK_T})$$

We now form:

$$\{g^{n_i}\}_{PK_T} \cdot \{h_i^{r_i}\}_{PK_T} = \{g^{n_i} \cdot h_i^{r_i}\}_{PK_T}$$

and verifiably decrypt this to give the trapdoor commitment $g^{n_i} \cdot h_i^{r_i}$. The g^{r_i} term is kept encrypted and secret.

6.3 Voting

Voter V_i casts her vote using a plaintext aware encryption scheme:

$$(\{\mathsf{Vote_i}\}_{\mathsf{pk_T}}, \Pi_\mathsf{i})$$

The plaintext awareness is needed to prevent an attacker copying, re-encrypting and casting a previously cast vote as his own. In conjunction with Selene such a copying attack would be particularly virulent: the attacker copies the victim's vote and casts it as his own, and when the votes and trackers are revealed he sees exactly how the victim voted. It is also advisable to post the votes only once voting has closed.

The eligibility and validity of ballots is checked and, if valid, the encrypted votes are posted to the PBB to give a list of tuples on the WBB:

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

6.4 Mixing and Decryption

Once voting has finished, for each row on the WBB, the validity of the ballot is checked for eligibility and well-formedness, according to the rules of the scheme. For valid ballots the pair of encryptions of the vote and the tracker are extracted and passed to the mixing process. 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 shuffles, e.g. [10] or using Verificatum. 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: $(n_{\pi(i)}, \mathsf{Vote_i})$.

6.5 Notification of Tracker Numbers

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 EA sends the voter V_j their share of the g^{r_i} over a private channel:

$$T_j \rightarrow V_i : \; g^{r_j}$$

The tracker commitment terms can be thought of as the second term of an ElGamal ciphertext, with the first term, the g^r term, kept hidden. On receipt of the α term the voter, or more precisely her device, can combine this with the tracker commitment term, the β term, to form the ElGamal encryption of her tracker under her trapdoor key.

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

The device can now perform the decryption, using the trapdoor key, and reveal the tracker.

Rather surprisingly, the α term is sent to the voter without any proof of origin or authenticity. The reason that we can do this is that an adversary with bounded computational power and not possessing the relevant trapdoor key, and even if colluding with all the Tellers, has only negligible probability of constructing an α term that opens up to a valid tracker different from the true tracker of the voter. Avoiding authenticating these notifications is more user-friendly because such communications have to be deniable and should be faked by the voter in case of coercion. Designated Verifier Signatures would be a way to sidestep such coercion threats, but they would significantly complicate the user steps in the event of coercion. The precise statement and proofs can be found in [13].

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. A possibility, suggested by D Wikström, is for the voter to send an encrypted blinding factor at the same time as casting the vote. This is then used to blind the α term when it is communicated to the voter.

6.6 Coercion: Threats and Mitigation

We have described how a voter can wrong-foot a coercer who demands that she reveal he tracker number, but what about a more insistent coercer who demands that she further reveal the alpha term, and that she demonstrate how this reveals the claimed tracker when input into her device. This is where the flip side of the construction comes into play: the voter, or more precisely her device, possessing the trapdoor key, can easily 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^*, she inputs this to her device along with the commitment value β_i and the device, with knowledge of the trapdoor key x, computes the fake term α':

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

This still leaves a coercion possibility: the coercer can demand to observe the receipt of the α. The α terms can be sent at randomized times forcing the coercer to monitor the voter's communications. However, the possibility of receiving the α term while the coercer is present, might still be discouraging for the voter.

A possibility to circumvent this is to provide a private channel to contact the voting authorities to request that the fake $(g^{r_i})'$ term that the voter has calculated be communicated back to her. This has spin-off effect of encouraging voters to notify the authorities of coercion.

6.7 Dispute Resolution

Dispute resolution, the ability for a judge to identify the cheating or malfunctioning component or party when an error is reported, is quite hard to achieve, especially in the internet voting context. Disputes could arise in performing Benaloh challenges for example: the audited ballot opens to reveal candidate A but the voter claims to have input B. There is no immediate way to distinguish between the device cheating or the voter lying, mis-typing or mis-remembering. It is essential that a well-designed system be able to make such distinctions reliably and in a fashion that can be proven. In the absence of such a property the system will be open to attacks attempting to discredit it or the election in question: e.g. many voters reporting fake complaints.

In Selene this could be tricky if a voter claims that the vote corresponding to their tracker is not what they cast. But this is a problem with the tracking number approach anyway. We could start to resolve this but encrypting the posted vote and tracker and performing a *plaintext equivalence text* against the cast encrypted vote and tracker appearing against the voter's Id before the mixing. If the tracker number do not agree this suggests that the voter is mistaken about her tracker. If the votes do not agree there has been a problem with the parallel mixing. If both agree, and the voter continues to insist that the vote is wrong, then it is possible that her device was corrupted and performed the encryption wrongly. This would all have to be performed with great care and suitable controls, and presumably *in camera* to avoid introducing coercion opportunities.

7 Selene II

We pointed out earlier that a coerced voter might have the misfortune of choosing the coercer's tracking number, or the coercer simply claims, falsely, that this is his tracker. In mild coercion threat contexts we may be able to ignore this issue, but if the threats of the coercer are sufficiently unpleasant this possibility could be enough to deflect the voter from voting her intent. The paper [13] provides a construction that provides voters with a set of alternative trackers, each pointing to one of the candidates, in such a way that these trackers are unique to her. If coerced she simply points to the tracker from this set that points to the coercers requested candidate, and now the coercer cannot claim ownership of this tracker. The tally board will now contain $c \cdot v$ additional tracking numbers, where c is the number of the candidates and v is the number of the voters. These will give one extra vote per candidate per voter which has to be subtracted in the tally. This is ok for simple plurality style elections, but not for more elaborate social choice functions, at least not without some adaptation. This aspect of the scheme is reminiscent of Rivest's *ThreeBallot* [12].

8 Conclusions

Democracy is under severe strain, and much of this arises from the introduction of digital technologies into elections, the media, social networks etc. In this paper I have focussed on just one small aspect of these threats: securing the casting and counting of votes. Modern cryptography and information security has made significant strides over the last decade or so in devising protocols and procedures to make the conduct of elections more transparent and auditable. These range from ensuring that all systems provide a well curated paper audit trail, enabling risk limiting audits, to the use of cryptographic techniques in E2E V schemes.

In particular I presented 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 prevents the usual coercer attack on such 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. Furthermore, the construction ensures a high level of assurance that the voter receives the correct tracker while ensuring that this is deniable.

The resulting scheme provides a very direct and simple to understand mechanism for voter verification while at the same time providing receipt freeness and mitigation of coercion. The crypto is kept under the bonnet for ordinary voters, and in particular the voter verification step involves checking 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 a more complex dispute resolution procedure. 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.

The Selene construction can be thought of as an add-on to existing non-verifiable schemes, 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 I scheme but if a significant level of coercion is reported before and during the vote casting period, the Selene II constructions could be dynamically added to the WBB give the higher degree of coercion resistance.

Note, Selene as presented here is intended for internet voting, but it would doubtless be straightforward to adapt it to in-person voting.

Acknowledgements. We would like to thank Sunoo Park and Bill Roscoe and many others for interesting discussions and suggestions. This work was partly supported by the FNR Luxembourg.

References

1. Adida, B., Neff, C.A.: Ballot casting assurance. In: Proceedings of the USENIX/Accurate Electronic Voting Technology Workshop, 2006 on Electronic Voting Technology Workshop (EVT 2006), Berkeley, CA, USA, p. 7. USENIX Association (2006)
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 Voting (EVOTE) (2012)
3. Benaloh, J.: Simple verifiable elections. In: Wallach, D.S., Rivest, R.L. (eds.) 2006 USENIX/ACCURATE Electronic Voting Technology Workshop (EVT 2006), Vancouver, BC, Canada, 1 August 2006. USENIX Association (2006)
4. Bursuc, S., Grewal, G.S., Ryan, M.D.: Trivitas: voters directly verifying votes. In: Kiayias, A., Lipmaa, H. (eds.) Vote-ID 2011. LNCS, vol. 7187, pp. 190–207. Springer, Heidelberg (2012). doi:10.1007/978-3-642-32747-6_12
5. Clarkson, M.R., Chong, S., Myers, A.C.: Civitas: a secure voting system. In: IEEE Symposium on Security and Privacy (2008)
6. Cortier, V., Fuchsbauer, G., Galindo, D.: Beleniosrf: a strongly receipt-free electronic voting scheme. IACR Cryptology ePrint Archive, 629 (2015)
7. 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). doi:10.1007/3-540-69053-0_9
8. Bell, S., et al.: Star-vote: a secure, transparent, auditable, and reliable voting system. In: 2013 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections (EVT/WOTE 13), Washington, D.C. USENIX Association (2013)
9. 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, 7 November 2005, pp. 61–70 (2005)
10. 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 (2010)
11. Randell, B., Ryan, P.Y.A.: Voting technologies and trust. In: IEEE Symposium on Security and Privacy, pp. 50–56 (2006)

12. Rivest, R.L.: The ThreeBallot voting system. https://people.csail.mit.edu/rivest/Rivest-TheThreeBallotVotingSystem.pdf
13. Ryan, P.Y.A., Rønne, P.B., Iovino, V.: Selene: voting with transparent verifiability and coercion-mitigation. In: Clark, J., Meiklejohn, S., Ryan, P.Y.A., Wallach, D., Brenner, M., Rohloff, K. (eds.) FC 2016. LNCS, vol. 9604, pp. 176–192. Springer, Heidelberg (2016). doi:10.1007/978-3-662-53357-4_12
14. Ryan, P.Y.A.: Pretty good democracy. In: Christianson, B., Malcolm, J.A., Matyáš, V., Roe, M. (eds.) Security Protocols 2009. LNCS, vol. 7028, pp. 131–142. Springer, Heidelberg (2013). doi:10.1007/978-3-642-36213-2_16
15. Ryan, P.Y.A., Schneider, S.A.: Prêt à voter with re-encryption mixes. Technical report CS-TR-956, University of Newcastle (2006)
16. Scantegrity Team. Scantegrity. http://www.scantegrity.org/papers/whitepaper.pdf
17. Stark, P.B., Lindeman, M.: A gentle introduction to risk-limiting audits. IEEE Secur. Priv. **10**, 42–49 (2012)
18. Wagner, D., Stark, P.B.: Evidence-based elections. IEEE Secur. Priv. **10**, 33–41 (2012)