Verifiable Postal Voting

Josh Benaloh[1], Peter Y.A. Ryan[2], and Vanessa Teague[3]

[1] Microsoft Research, Redmond, WA, USA
benaloh@microsoft.com
[2] University of Luxembourg
peter.ryan@uni.lu
[3] Department of Computing and Information Systems,
University of Melbourne, Australia
vjteague@unimelb.edu.au

Abstract. This proposal aims to combine the best properties of paper-based and end-to-end verifiable remote voting systems. Ballots are delivered electronically to voters, who return their votes on paper together with some cryptographic information that allows them to verify later that their votes were correctly included and counted.

We emphasise the ease of the voter's experience, which is not much harder than basic electronic delivery and postal returns. A typical voter needs only to perform a simple check that the human-readable printout reflects the intended vote. The only extra work is adding some cryptographic information into the same envelope as the human-readable vote.

The proposed scheme is not strictly end-to-end verifiable, because it depends on procedural assumptions at the point where the ballots are received. These procedures should be public and could be enforced by a group of observers, but are not publicly verifiable afterwards by observers who were absent at the time.

Keywords: electronic voting, verifiability, postal voting, vote by mail, end-to-end verifiable voting.

1 Introduction

There are no good options for voters unable to visit a polling place. Snail mail is slow, unreliable and easily intercepted, but it has one great advantage: ordinary people can see clearly what they have sent. This is the same advantage that has made a human-readable paper trail a focus of attempts to improve the integrity of polling-place DRE voting machines. Voters all over the world are clamouring for a substitute for postal voting, with its numerous inconveniences. Postal voting is also much less secure than attendance paper voting, being more susceptible to both privacy compromise and vote manipulation. It struggles to satisfy fast delivery requirements, in two directions, over what can be a very slow channel. Many people who haven't thought much about electronic security think that Internet voting is a great alternative. It's a pity about the human-readable paper record though.

B. Christianson et al. (Eds.): Security Protocols 2013, LNCS 8263, pp. 54–65, 2013.

One obvious improvement to postal voting is to deliver ballot information electronically and then ask voters to return paper votes by mail [JS12]. This cuts out the difficult half of the snail-mail delivery, and provides simple cast-as-intended verification, but it gives no more guarantees than ordinary postal voting of privacy, delivery, or accurate counting.

An alternative approach is to use cryptography to mitigate the vulnerabilities of (Internet-based) electronic voting. End-to-end verifiable systems such as Helios [Adi08] provide proofs that all the votes were counted as cast and correctly tallied. The most difficult part is allowing voters to verify that their votes were cast as they intended, even given the possible presence of malware on their computers. Helios voters are encouraged to perform a randomised protocol to test whether the vote is recorded in the way they intended. If the voters perform the protocol correctly, they get very good evidence that their vote was cast as they intended. It isn't necessary for all voters to perform this check—the important point is that a manipulating machine risks detection unless it can be confident the voter won't check. There is therefore a key assumption that regardless of the voter, a Helios client will never have prior certainty that the voter will not perform a check. Whether this assumption is valid depends, at least in part, on the population of voters. The IACR election is highly likely to have a large enough population of sophisticated voters that any cheating attempt has a high probability of detection. However, for some ordinary voters in government elections, it could be much easier for a malicious machine to predict that the voter will not check, or to trick them into not checking.

Our proposal is to try to get the best of both worlds, with a simple cast-as-intended check for most voters and a verifiable protocol demonstrating correct inclusion and counting. The scheme uses both an electronic and a snail-mail channel — blending the best properties of each.

We emphasise the ease of the voter's experience, which is not much harder than basic electronic delivery and postal returns. The extra cryptographic information needs only to be added into the same envelope as the human-readable vote. Mechanisms for assisting voters with disabilities could easily be incorporated into the process of filling in the ballot by computer (though not quite so easily into the process of putting the printouts in an envelope and posting it). The proposal provides a set of security properties not obtainable on other remote systems with such an easy voting experience.

The proposed scheme is not strictly end-to-end verifiable, because it depends on procedural assumptions at the point where the ballots are received. These procedures should be public and could be enforced by a group of observers, but are not publicly verifiable afterwards by observers who were absent at the time.

1.1 Related Work

A completely different approach is Code Voting [Cha01], in which voters receive a code sheet in the mail and then use codes to communicate their choices to the electoral authorities via an untrusted electronic device, or to check via a return code that the authorities received the correct choice. This style of remote voting has been used in government elections, for example in Norway [Gj10]. Chaum's original code voting scheme assumed an honest electoral authority. Although subsequent works have weakened this assumption substantially, all code voting schemes still require a secrecy assumption for the integrity of the election. In other words, a malicious device that learns the codes can manipulate the vote. PGD [RT13] allows the code information to be generated in a distributed way, and Remotegrity [ZCC$^+$13] uses physical protections such as scratch strips to protect the data in transit, but there are still strong assumptions about the security of the (postal) delivery system. Furthermore, code schemes are difficult to use when the ballot is complex, such as in IRV/STV elections with many preferences [HRT10]. Our proposed system works for any voting or tallying scheme. One interesting difference is that code-based schemes send out a piece of paper and then receive the vote electronically, while our proposal sends ballot information electronically and then requires a paper return.

Two polling-place voting systems elegantly combine human-readable paper records with end-to-end verification. In the Wombat voting system [RTsRBN], voters produce both human-readable and encrypted versions of their votes. Because Wombat is an attendance voting system, the process of reconciling and then separating the two representations is performed by the voter, using a process similar to Benaloh's simple verifiable elections [Ben06]. In the StarVote system proposed for Austin, TX, [BBK$^+$12], voters make both human-readable plaintext representations and encrypted representations, check that the former matches their intentions, and then either cast or audit their ballots. In addition, the plaintext representation is part of a risk-limiting audit in the style of SOBA [BJL$^+$11]. Our question is how to achieve a similar set of security guarantees in a remote setting without asking the voters to do too much work.

Our proposal uses a different method of combining the benefits of cryptographic-style verification with randomised, publicly-observable checking of paper records. The rough idea is that each voter produces a human-readable paper record, and a (non-human-readable) encrypted record. Voters check that the former matches their intention, and submit the latter into a process of public auditing which verifies that with high probability the encrypted records match the paper ones. The following section describes the background assumptions. We then provide an overview of the protocol, followed by some important details, then a discussion of some possible variants.

2 Assumptions and Requirements

Some simple assumptions about voting:

1. *The electoral authorities maintain an accurate list of who is eligible to vote,*
2. *There is a public list linking a public key to each eligible voter.*

Some more complex assumptions about postal voting in particular:

3. *There is sufficient observation or proper process at the vote receiving location to ensure that votes are not lost upon arrival and some observable procedures are followed.*

We will require an unpredictable coin toss for each received vote. The trick is to design a process that provides good counted-as-cast evidence to observers. See below.

4. *There is an irrevocable process for separating pieces of paper that arrive in the same envelope, from each other and from the envelope.*

   This process varies somewhat from one country to another, but is used to separate the voter's identity, usually on an external envelope, from the contents of the vote, usually inside an(other) envelope. Traditionally the vote is mixed in a box with other votes. We will also use this to separate irrevocably pieces of paper that originated in the same envelope.

Like all end-to-end verifiable voting protocols, we assume we have a Bulletin Board, which is an electronic authenticated write-only broadcast channel with memory ("broadcast" means that everyone is guaranteed to see the same data). Some requirements for the system:

1. *Vote privacy.* It should be infeasible to link individual voters to their vote. However the system is not receipt-free (and hence not coercion resistant).
2. *Eligibility Verifiability.* The list of public keys of admitted voters is public.
3. *Cast-as-intended (individual) verifiability.* Voters should each have evidence that their votes were cast as they intended.
4. *Counted-as-cast verifiability.* Each observer should each have evidence that all votes were counted as they were cast. (Note that in end-to-end verifiable systems this is verifiable by voters; here it is verifiable by any observer who participates in the vote-opening protocol.) We have two different variants with different assumptions for counted-as-cast verifiability—see Section 3.3.
5. *Universally verifiable tallying.* Voters and observers alike can verify the correct tallying of all cast votes.

The scheme aims to defend against:

1. *An attacker who manipulates paper votes in transit.*
   This should be detected at audit time with probability at least 1/2.
2. *A malicious voting device that misprints the plaintext paper record.*
   This should be detected by the voter.
3. *A malicious voting device that manipulates the encrypted record*
   This should be detected at audit time with probability at least 1/2.
4. *A collusion of some of the electoral authorities opening envelopes and their observers.*
   One honest observer should be able to detect departure from the protocol.

It does not aim to defend against an attacker who drops postal votes—this can be detected, but cannot be distinguished from "honest" failures of the mail system. Nor does it defend against a complete collusion of all electoral authorities and all of the observers at envelope-opening time. In other words, at least one observer must be honest. Although the scheme defends against either a malicious voting device or an attacker who controls the postal voting channel, it is susceptible to collusion between those two attackers. There is also a strong assumption that the paper records, once received, are properly secured.

3 The Proposal

Voters receive voting information, such as candidate names, electronically. They fill out their vote on their computer, and print three representations of it on separate pieces of paper:

- *VR*, a human-readable plaintext vote,
- *VE*, an encrypted and signed representation of the vote, which identifies (in non-human-readable form) whose key it is signed with.
- *VI*, an encrypted but not signed representation of the vote, which is printed along with two large random values:
  - *RI* is the random value used to encrypt *VR* to produce *VI*,
  - *RE* is the random value used to re-encrypt *VI* to produce *VE*.

  (These values may or may not be encrypted—see below.)

All three printouts go in the same postal envelope.

Each voter retains a copy of *VE* as a receipt (in printed or electronic form), but voters' computers are supposed to delete *VI* and the random values used to produce *VI* and *VE*.

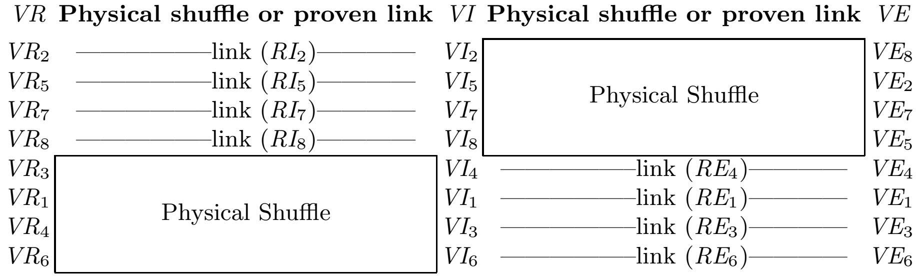
The difficult part is to allow any observers present at the opening of the voting envelopes to get evidence that the votes are counted as cast, without compromising privacy. In other words, by checking that for each vote, *VR* and *VE* represent the same thing. We will do this by using the intermediate representation *VI*. Observers randomly choose whether to get evidence that *VI* matches *VR* or that *VI* matches *VE*. (This evidence consists of learning either *RI* or *RE*). The implementation details could vary with the voting scheme and the requirements of the paper delivery mechanism.

The result is similar to Randomised Partial Checking [JJR02]. On the bulletin board go the complete list of plaintext votes $VR_1, VR_2, \ldots$ the complete list of intermediate representations $VI_1, VI_2, \ldots$ the complete list of signed, encrypted votes $VE_1, VE_2, \ldots$, and, for each vote, either a value (RI_i) proving the link from VR_i to VI_i, or a value (RE_j) proving the link from VI_j to VE_j. Like RPC, privacy is reasonable but imperfect: each vote is anonymised among half of the set. See Figure 1.

The difficulty is to design an easy process for publishing (and proving) either the link from *VI* to *VE* or the link from *VI* to *VR*, while hiding or destroying the other link.

3.1 Details 1: How One Link Can Be Published and the Other Destroyed

One possibility (from now on called *the crypto option*) is to encrypt both *RE* and *RI*, print both encrypted values on the same piece of paper as *VI*, post both encrypted values on the Bulletin Board, and then decrypt only the one that is selected. The other value remains encrypted and hence does not reveal the link between *VI* and the other data item (*VR* or *VE*).



VR	**Physical shuffle or proven link**	*VI*	**Physical shuffle or proven link**	*VE*
VR_2	link (RI_2)	VI_2	Physical Shuffle	VE_8
VR_5	link (RI_5)	VI_5		VE_2
VR_7	link (RI_7)	VI_7		VE_7
VR_8	link (RI_8)	VI_8		VE_5
VR_3	Physical Shuffle	VI_4	link (RE_4)	VE_4
VR_1		VI_1	link (RE_1)	VE_1
VR_4		VI_3	link (RE_3)	VE_3
VR_6		VI_6	link (RE_6)	VE_6

Fig. 1. Information published on the Bulletin Board to demonstrate votes are counted as cast

A second possibility (from now on called *the paper option*) is to use physical paper mechanisms to separate and destroy the unused value. For example, *RI* and *RE* could each be printed on its own separate piece of paper. The selected value could then be attached to *VI* and published, while the unselected value was shredded. Alternatively, both values could be printed on the same piece of paper as *VI*, but the unused one could be detached and shredded.

These two options seem to achieve the same effect. The first option involves more cryptographic work; the second involves more fiddling with pieces of paper. The structure of the protocol is the same in each case.

3.2 The Rest of the Protocol

When the envelopes arrive at the electoral authority:

1. For each envelope, the signature on *VE* is verified without revealing to observers whose signature it is.
2. For each envelope, a coin is tossed which determines whether observers will later get a link from *VR* to *VI*, or a link from *VI* to *VE*. The piece of paper containing *VI* is accordingly stapled to either *VE* or *VR*, depending on the coin toss.

   If we are using the paper option, the appropriate random value (*RE* proving *VI* matches *VE* or *RI* proving *VI* matches *VR*) must also be stapled to *VI*, while the other value is shredded. With the crypto option both encrypted values and *VI* are printed on the same piece of paper and don't need special treatment here.
3. When all envelopes are opened there are four (nearly) equally large piles of paper:
   (a) *VR* with *VI* and (possibly encrypted) *RI* stapled to it,
   (b) *VR* without *VI* stapled to it,
   (c) *VE* with *VI* and (possibly encrypted) *RE* stapled to it,
   (d) *VE* without *VI* stapled to it.
   Each is shuffled in an ordinary ballot box, then retained as evidence.
4. The pile of *VR* with *VI* stapled to it has its *RI* values (proving the equivalence of *VI* and *VR*) published on the Bulletin Board. If we are using the crypto

option the encrypted value is published, then provably decrypted.[1] (The unused value doesn't need to be published, and should never be decrypted.)
5. The pile of *VE* with *VI* stapled to it has its *RE* values (proving the equivalence of *VI* and *VE*) published on the Bulletin Board. The crypto option is the same as for the *VR* with *VI* pile.
6. The other data contained in the other two paper piles (the lists of *VR* and *VE* unlinked to their corresponding *VI*) is also published on the Bulletin Board.
7. The list of encrypted, signed votes are cryptographically mixed (or homomorphically tallied) and verifiably decrypted.

Everyone can verify the proofs, but of course the cast-as-intended evidence depends on the coins being properly tossed so that the match of *VR* with the electronic data is verified.

Some care needs to be taken for the first of the above steps, *i.e.* verifying the signatures at envelope-opening time without revealing to the observers whose signature it was, or to the (electronic) signature reader which vote it was. Ensuring this separation is crucial for privacy, and has to be enforced procedurally. The signature would be in a format (such as a QR code) that's prohibitively difficult for humans to read or remember by sight. The electronic signature reader would scan only the signature on *VE*, and the observers would be forbidden from pointing technological devices at the ballots.

3.3 Details 2: How the Random Bit Selection for Each Ballot Should Be Performed

There are various sources of randomness. It's important that the source is unpredictable to whichever attacker tries to manipulate the vote. Here are two example sources:

The Voter, Using a Combination of the Electronic and Paper Channels. Individual cast-as-intended and counted-as-cast verifiability could be achieved by having voters themselves make the "random" selections as to which of *RI* or *RE* will be revealed after their ballots have been printed. This could be accomplished by explicitly indicating which of the two links should be revealed (perhaps by ticking a separate box). Dilligent voters could remember which of *RI* and *RE* they had selected and see on the subsequent public postings that the correct one had been revealed.

One problem with this apporach is that humans are notoriously poor at making random selections, and this would need to be accounted for along with cases where no selection is made.

[1] It isn't entirely clear that a proof of correct decryption is necessary here given that we've assumed the paper trail is properly guarded after being received at the electoral commission. However, it seems important not to introduce an opportunity to pretend it matched a value different form what the voter saw.

A physical method of obtaining randomness from voters could work well here. The ordering or orientation in which the ballot paper(s) are placed in the envelope could be used as a source of random selections. (There are at least four distinct and easily distinguished orientations in which a — folded or unfolded — rectangular piece of paper can be placed in a slightly larger rectangular envelope. Two simple classes can be whether the leftmost printing of the front of the ballot is placed against the leftmost or rightmost printing of the front of the envelope.) Since every ballot must be oriented somehow within the envelope, there would always be a "random" selection — presumably not known in advance by the voter's computer. Knowledgeable voters could take note of this orientation and check that the correct value is subsequently revealed.

However, this approach has one important weakness: an attacker who controls the postal system (and can open envelopes and reseal them) can see which *VR* records will not be checked against *VI*. This allows the possibility of substituting *VR* undetectably. (A full mix and decryption of *VE* records will detect the anomaly—it just won't be clear what caused the problem, or which result is correct. See Section 5 for a longer discussion.)

It is preferable to ask the voter to send one bit in the paper envelope, and a separate bit electronically (via the same machine that they use for vote printing). The checking of *VI* against *VR* or *VE* could then be chosen by taking the XOR of that voter's two bits. The aim would be to prevent the machine from learning the "paper" bit (and hence manipulating the electronic bit and the encrypted records), and prevent anyone who intercepted the paper record from knowing the electronic bit (and manipulating the paper bit and paper record).

Voters could access their *VE* records and verify that the correct links had been opened. This reduces the trust assumptions on the observers to not knowing the electronic bit in advance, and not manipulating the paper records afterwards.

The Observers or Electoral Authorities Jointly, Using Jointly Generated Data and Data from the Vote. We could do a more traditional distributed randomness generation, either using cryptographic joint coin-tossing or the sort of machine used in lotto. In this case we're assuming that at least one observer at the receiving end honestly inputs some randomness, and there's a commonly available PRNG to expand the seed into a string of random bits.[2] This could be applied to ballots in some predetermined order, or combined with some randomness generated from the ballot itself. There are two options:

- **ballot order:** The ballot order would be fixed in advance, or drawn at random. The seed would be used to generate a pseudorandom string which was applied to each ballot choice in turn.
- **ballot contents:** The bit would be (part of) the output of a hash of both the seed and some data on the ballot. One possibility is to use only data

[2] One concrete possibility is to use Stark's tools for generating randomness for risk-limiting audits, available at `http://www.stat.berkeley.edu/~stark/Java/Html/ballotPollTools.htm`

from *VE*, in which case anyone can use the data on the Bulletin Board to check that the correct link has been opened for each vote.

This reduces the trust assumptions on the observers to not arranging the envelopes with knowledge of their contents, and not manipulating the paper records afterwards.

Comparison of Approaches to Random Bit Selection. One way to compare the proposals is to think about an attacker with unsupervised access to a ballot at different times. Our attacker wants to substitute *VR*.

- If the attacker has the ballot before the envelope has arrived at the vote receiving location, and before the seed has been generated, then (with either method) the attack has at least a 1/2 chance of being detected.
- If the attacker has the ballot after the seed has been generated, and before it arrives at the vote receiving location (or before it's been properly accepted into a secure storage area), then with the "ballot contents" scheme the attacker knows which half of the ballots can be safely manipulated; with the "ballot order" scheme the attacker also has to arrange for the ballot order to be manipulated.
- If the attacker has the ballot before the seed has been generated, then with the "ballot contents" scheme the attack will still be detected with probability 1/2. With the "ballot order" scheme, the attacker will (still) need to collude with someone who manipulates the ballot order.
- An attack on the scheme where the voter chooses two bits has at least a 1/2 chance of being detected, assuming that the electronically-sent bit is secret and was randomly generated. (But this is possibly a too-strong assumption given that people are not good at choosing random values or keeping secrets.)

The crucial point with the keyed scheme is not to generate the seed until all the ballots are in, past the point where they're subject to manipulation. One option is to generate a new seed every day.

4 Privacy

Since voters mark their ballots electronically, there is no defence against eavesdroppers or malware such as keyloggers resident on the voter's computer system. The system otherwise provides reasonable (though not perfect) privacy but is not receipt-free. (We could encrypt the signature and the voter's ID so that only the electoral authority could identify whose it was. This would mitigate eavesdropping on the snail mail and not otherwise affect the protocol, except that it would require an additional decryption step when the vote arrives.)

When the envelopes are opened, all of the vote and identification data are present together. At the time the signature is verified electronically, the vote information is not supposed to be available to the electronic system. When the human-readable paper vote is exposed, the observers are not supposed to learn the identity of the signature that is being verified. Both of these need to be enforced by procedural mechanisms as the envelopes are being opened and the pieces of paper stapled together. Similarly, the proper shuffling of each of the four piles of paper is necessary for breaking the links between corresponding elements.

Voters are obviously supposed to erase *VI* and its associated randomisation/re-encryption factors. If they remember this data then they can prove how they voted. This is why the protocol is not receipt-free.

5 Verifiability

Cast-as-intended verifiability is achieved very straightforwardly by letting each voter print a human-readable vote *VR* and check it before placing it in the envelope. Universally verifiable tallying is achieved by publishing the paper votes in cleartext, so they can be tallied directly. Voters check that their electronic records have been properly received by looking up *VE* on the Bulletin Board.

Counted-as-cast verifiability consists of checking that *VE* matches *VR*. This is done by allowing observers to choose randomly whether to get a proof of *VI* matching *VE* or *VR*. Of course the quality of this assurance depends on the randomness. If the two options are chosen randomly and independently then the probability of successfully manipulating votes decreases exponentially with the number of manipulations. We have described two different proposals which give evidence of proper random generation to different sets of observers. One gives each individual voter control over their own random bit selection; the second gives a group of observers evidence about the proper bit selection of the collection of votes.

This system could be have been designed as two independent partially-verifiable systems: a simple paper system of electronic ballot delivery and (human-readable) paper returns, plus a (non-human-readable) computerised system in which the voter can use cryptography to verify proper inclusion and tallying, but not that their vote was cast as they intended. We could have simply compared the paper count to the cryptographically verifiable electronic tally and declared success if they matched. Numerous cryptographic schemes exist that are truly universally verifiable (*e.g.* [CGS97], [SK95]), and ensure that the probability of a single undetectable vote substitution by the authorities would be exponentially small. However, there is no cast-as-intended verification: if the electronic tally differed from the paper records, it would not be clear whether the paper record had been manipulated, or a malicious voting computer had sent the wrong vote.

The problem, of course, is that in any practical election they'd be unlikely to match perfectly, and it would be impossible to understand what had gone wrong. This would raise unanswerable questions about which tally to accept—the answer would depend on a guess about what had caused the inconsistency.

The auditing step suggested here, in which each ballot is checked for consistency between its encrypted and human-readable versions, ensures that the paper and electronic counts are very unlikely to differ by much. This should obviate the need to decide whether it's the paper count or the electronic count that's the "true" count.

There is still a firm assumption about proper care of the physical paper evidence, particularly the half that's not cryptographically linked to *VI*. This seems unavoidable with a simple VVPAT.

Although the scheme defends against an attacker who controls either the voting device or the postal channel, it does not defend against colluding attackers who control both. The malicious device could print a human-readable record that matches the voter's intention, but encrypted electronic records for a different choice, then the attacker could switch the human-readable record in the mail.

6 Other Variants and Discussion

It would be possible to ask voters to send *VE* electronically to the electoral authorities (as well as the printed version). This increases the complexity, but also has the benefit that the count could commence much sooner. It could be possible to mix and tally electronically "optimistically," meaning that the electronic record would be used, but the paper records would subject to audit in close races, or kept in case of a dispute.

One important practical complication is that some electronically delivered votes will not subsequently appear in paper form, due to failure of the mail. It's unclear what to do in this situation, but the simplest defensible thing is not to count them. (The alternative is to count them anyway, but then there is no cast-as-intended verifiability.) Hence the authorities must at least open each envelope and check which votes have arrived.

Another design direction worth investigating is to attempt to achieve everlasting privacy [MN10] by using perfectly hiding commitments rather than encryptions of *VI* and *VE*. This would mean that integrity depended on a computational assumption (that a computationally binding commitment could not be opened in more than one way), but this could be a reasonable tradeoff, especially since integrity depends on distributed randomness generation and associated procedures here anyway. It would require a way of either adapting or omitting the electronic tallying step.

7 Conclusion

This system makes strong, but observable procedural assumptions for both verifiability and privacy, but almost all parts of the process are individually or universally verifiable. This represents a reasonable tradeoff among the conflicting requirements of remote voting.

References

[Adi08] Adida, B.: Helios: Web-based Open-Audit Voting (2008)

[BBK$^+$12] Benaloh, J., Byrne, M., Kortum, P., McBurnett, N., Pereira, O., Stark, P.B., Wallach, D.S.: Star-vote: A secure, transparent, auditable, and reliable voting system (2012)

[Ben06] Benaloh, J.: Simple verifiable elections. In: Proc. 1st USENIX Accurate Electronic Voting Technology Workshop (2006)

[BJL$^+$11] Benaloh, J., Jones, D., Lazarus, E., Lindeman, M., Stark, P.B.: Soba: Secrecy-preserving observable ballot-level audit. In: Proc. USENIX Accurate Electronic Voting Technology Workshop (2011)

[CGS97] 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)

[Cha01] Chaum, D.: SureVote: Technical Overview. In: Proceedings of the Workshop on Trustworthy Elections, WOTE 2001 (2001)

[Gj10] Gjsteen, K.: Analysis of an internet voting protocol. Cryptology ePrint Archive, Report 2010/380 (2010), `http://eprint.iacr.org/`

[HRT10] Heather, J., Ryan, P.Y.A., Teague, V.: Pretty good democracy for more expressive voting schemes. In: Gritzalis, D., Preneel, B., Theoharidou, M. (eds.) ESORICS 2010. LNCS, vol. 6345, pp. 405–423. Springer, Heidelberg (2010)

[JJR02] Jakobsson, M., Juels, A., Rivest, R.: Making Mix Nets Robust for Electronic Voting by Randomized Partial Checking. In: USENIX Security Symposium, pp. 339–353 (2002)

[JS12] Jones, D.W., Simons, B.: Broken Ballots: Will Your Vote Count? University of Chicago Press (2012)

[MN10] Moran, T., Naor, M.: Split-ballot voting: Everlasting privacy with distributed trust. ACM Transactions on Information and System Security 13, 16:1–16:43 (2010)

[RT13] Ryan, P.Y.A., Teague, V.: Pretty Good Deomcracy. In: Proceedings of the Seventeenth International Workshop on Security Protocols 2009 (2013)

[RTsRBN] Rosen, A., Ta-shma, A., Riva, B., Ben-Nun, J.(Y.): Wombat voting system

[SK95] Sako, K., Kilian, J.: Receipt-free mix-type voting scheme. In: Guillou, L.C., Quisquater, J.-J. (eds.) EUROCRYPT 1995. LNCS, vol. 921, pp. 393–403. Springer, Heidelberg (1995)

[ZCC$^+$13] Zagórski, F., Carback, R.T., Chaum, D., Clark, J., Essex, A., Vora, P.L.: Remotegrity: Design and use of an end-to-end verifiable remote voting system. In: Jacobson, M., Locasto, M., Mohassel, P., Safavi-Naini, R. (eds.) ACNS 2013. LNCS, vol. 7954, pp. 441–457. Springer, Heidelberg (2013)