Practical Governmental Voting with Unconditional Integrity and Privacy

Nan Yang and Jeremy Clark(✉)

Concordia University, Montreal, Canada
na_yan@encs.concordia.ca, j.clark@concordia.ca

Abstract. Throughout the years, many cryptographically verifiable voting systems have been proposed with a whole spectrum of features and security assumptions. Where the voter casts an in-person (and possibly paper) ballot and leaves, as is common in a governmental election, the majority of the proposals fall in the category of providing unconditional integrity and computational privacy. A minority of papers have looked at the inverse scenario: everlasting privacy with computational integrity. However as far as we know, no paper has succeeded in providing both unconditional integrity and privacy in this setting—it has only been explored in boardroom voting schemes where voters participate in the tallying process. Our paper aims for a two-level contribution: first, we present a concrete system with these security properties (one that works as a backend for common ballot styles like Scantegrity II or Prêt à Voter); and second, we provide some insight into how different combinations of security assumptions are interdependent.

1 Introduction

An end-to-end verifiable (E2E) voting system uses cryptography to provide a verifiable tally while maintaining the secrecy of each voter's ballot. Over decades of research in this area, one trend to emerge is a move toward real-world voting systems suitable for common election scenarios, including governmental elections. For our purposes, we consider a system to be suitable for a governmental election if it has two properties:

1. **Vote-and-go**: once a voter has completed and submitted their ballot, they do not need to be involved in the tallying process.
2. **Human-votable**: a voter can cast a vote without having to perform any computations (bare-handed) through a process similar to a traditional (non-verifiable) voting system, such as DRE or optical scan voting

Many E2E systems are designed within these constraints and some have been used in governmental elections [6,7]. The governmental setting is contrasted with other practical settings, such as a boardroom vote, where all voters might be physically present in the same room with their own trusted computational devices. This setting is less constrained and allows different cryptographic techniques to be used—e.g., an unconditionally secure multiparty computation.

M. Brenner et al. (Eds.): FC 2017 Workshops 2017, LNCS 10323, pp. 434–449, 2017.
https://doi.org/10.1007/978-3-319-70278-0_27

In the governmental setting, vote-and-go requires a third party election authority to collect a representation of the voter's ballot. This representation is often an encryption or commitment to the voter selections for DRE-based systems, or for optical scan systems, a paper-based obfuscation (e.g., code substitution, permutation, split) that is accompanied by some encryption or commitment value on the ballot or in the backend data. Standard encryption and commitment schemes are not secure against a computationally unbounded adversary. Such an adversary can either recover the message (Elgamal or Paillier), change the message (Pedersen commitment) or both (hash-based commitments). When the message is a vote, this translates into, respectively, breaking election integrity or ballot secrecy or both.

Computational assumptions underly nearly all real-world cryptographic applications, whether it is HTTPS, password hashing, or secure messaging. However the exact assumptions evolve over time as new attacks are found, as do the security parameters that realize them. An unconditionally secure protocol alleviates us from monitoring the validity of these assumptions over time and future-proofs the protocol against new innovations like quantum computing.

2 Prior Work

There are hundreds of papers proposing voting schemes and it is not possible to review even all the relevant ones. Instead, we have broken the literature into four broad categories that classify a majority of the proposals. Table 1 provides a summary of the election integrity and ballot secrecy assumptions for each cluster.

Distributed EA. Beginning with Cramer et al. [16], many systems homomorphically encrypt ballots under a public key that is distributed amongst a set of trustees forming an election authority (EA). If an unbounded adversary attacks a transcript of the election, they can learn how every voter voted by breaking the encryption key but cannot change the value that is encrypted. Further, assuming true zero knowledge proofs are used, unbounded adversaries cannot undetectably change the tally. Note that in practice, many of these systems use non-interactive zero knowledge proofs based on the Fiat-Shamir heuristic — this enables an unbounded adversary (whether a voter or a trustee) to lie [21] in a way that can undetectably change a tally, however this assumption is practical to avoid [19,26]. If a suitable threshold of trustees are corrupted, they may recover how each voter voted but they cannot change the tally. A few notable systems of this type include: MarkPledge [33], Prêt à Voter [12], Voter-initiated auditing [3], Helios [1], STAR-Vote [2], and vVote [6].

Chaumian. Beginning with Chaum [9], a series of systems also use a distributed election authority much like above. However these systems add an additional assumption: trustees can use a special computational device, called a blackbox, to perform computations such that the inputs and intermediate values are not leaked to any participant. This enables an election system based solely

Table 1. A comparison of computational and collusion security assumptions in four common categories of proposed cryptographic voting systems, plus our own system. Note: this table does not attempt to capture all desirable features of a voting system. We acheive the same security assumptions as boardroom voting systems, plus we allow human-voteable ballots and vote-and-go tallying. The 'special assumption' used in Chaumian systems (and this work) is a blackbox assumption.

Category	*Examples*	*Secrecy*		*Integrity*		
				Resilient to Unbounded Adversary	Resilient to Full Collusion	No Special Assumptions
Distributed EA	Pret a Voter, Helios		•	•	•	•
Chaumian	Punchscan, Scantegrity			•	•	•
Everlasting Privacy	Moran-Naor	•	•		•	•
Boardroom	Broadbent-Tapp	•	•	•		•
This work		•		•		•

on cryptographic commitments and commitment-based cut-and-choose proofs. Assuming the commitment scheme is perfectly binding, an adversary can break ballot secrecy by breaking the commitment scheme (if unbounded), corrupting a sufficient number of trustees to recover the input to the blackbox, or by breaking the blackbox hardware assumption. However an unbounded adversary cannot undetectably change the values committed to, all modifications to the tally are detectable even if made by a fully colluding election authority, and the soundness of the blackbox computations are verifiable and not assumed to be done correctly. Notable systems of this type include Punchscan [34], Scantegrity I/II [10,11], Eperio [18], and Remotegrity [41].

Everlasting Privacy. Beginning with Cramer et al. [15] (and related to earlier work by Chaum [8]), a reasonable observation was made that integrity need only last the lifetime of the election but ballot secrecy could be relevant for decades or centuries. It is possible to invert the resistance of a voting scheme to computationally unbounded adversaries from integrity to privacy. Most modern work uses perfectly hiding homomorphic commitments in lieu of homomorphic encryption, however this creates a dilemma: if the random factors of the commitments are unknown, a tally cannot be computed (and if they are known, then the commitment's hiding property no longer resists an unbounded adversary). Most systems compromise by using untappable channels to communicate random factors amongst trustees— thus it does not retain unconditional ballot secrecy under collusion. Notable systems of this type include Moran-Naor [31],

split-ballot voting [32], and extensions to distributed EA systems [17]. Recent work from Locher et al. has examined the removal of the collusion assumption, presenting schemes [27,28] that have everlasting privacy under *both* an unbounded and fully colluding EA (with computational integrity).

Boardroom Voting. The term boardroom voting was suggested by Benoloh and Fisher [4] to categorize systems where voters participate in the tallying process (i.e., are not vote-and-go). Like the general literature on unconditionally secure protocols, these schemes tend to use multiparty computation based on verifiable secret sharing. Note that not all boardroom voting schemes are unconditional—many boardroom systems use computational assumptions to be more practical [22,25,38,39]. However the ones that are resist unbounded adversaries for both integrity and privacy (but collusion between them can break either property). One way to frame our contribution is porting the security properties of these systems to a governmental election. This has been explored [5] and the *vote-and-go* property is achieved, voters need to perform computations in the booth (and it is thus not *human voteable*). One might argue that ThreeBallot [37] is a human-voteable instantiation of secret sharing. Indeed, its properties are very close to what we want to achieve. Unfortunately ThreeBallot is not fully private [23].

3 Framing Our Contribution

It has long been asserted within our community that perfect ballot secrecy and perfect election integrity cannot be simultaneously achieved. This trade-off is quite true under certain assumptions but it is often repeated as a simple fact without internalizing the fine print. As it turns out, if you read the fine print, it is possible to achieve both—indeed many boardroom voting systems already do. The challenge is achieving these security properties while also allowing the voter to deposit their ballot with the EA and leave. If the deposited ballot is an encryption or computational commitment, it must be either computationally binding or hiding but not both. If the ballot is secret shared to the trustees, however, it can be perfectly hiding and binding under an assumption about the number of honest trustees. The immediate difficulty here is that secret sharing a vote will require a computational device.

This paper is intended as exploratory research to understand better how far unconditional privacy and integrity can be extended to a practical governmental voting system. We are not insisting that our system is immediately better than existing approaches because we require certain trade-offs that might be less desirable (discussed below). However we think this area deserves exploration.

In our approach, we begin in the Chaumian model. We noted in our literature review that systems in this model primarily rely on a commitment scheme. As we discuss in Sect. 4.1, verifiable secret sharing can be used as a perfectly hiding commitment that is also perfectly binding but only to the participants in the secret sharing scheme. We take a simple system from this model, Eperio [18], which is already just a backend tallying system that can interface with a variety

of paper ballots (permutation-based ballots like Prêt à Voter and code-based ballots like Scantegrity), and we replace the commitment scheme with a protocol based on verifiable secret sharing. We then show that the cut-and-choose protocols continue to provide election integrity, assuming an honest threshold of trustees (which is already assumed in computational Eperio for ballot privacy). The result is an interesting protocol that achieves unconditional privacy and integrity, plus voters can vote with paper ballots.

Universal verification. We pay a price for unconditional secrecy and privacy, namely we have to sacrifice universal verification. Chevallier-Mames et al. prove that achieving unconditional privacy is sufficient to thwart universal verification (if it is possible for voters to choose to abstain from voting) [13]. Under slightly different definitions, Vora and Hosp show an impossibility 'triangle:' it is only possible to achieve two of the three properties: perfect integrity, perfect privacy, and universal verification [24]. They define integrity and privacy in an information theoretical sense. We also note that attempts of adding it to the basic primitive we use (VSS) generally has only been achieved with computationally secure primitives [38,40]. In our protocol, voters can still perform the traditional *cast-as-intended* and *recorded-as-cast* checks but voters have to trust that a threshold of trustees are honest in reporting that ballots were *tallied-as-recorded*. It is not clear this trade-off is worth the gain in security against unbounded adversaries, but we will say that it is not that different from cryptographic election where voters defer to others (say each political party) to perform the cryptographic election audit of the tally. Finally, our approach of using paper ballots does not preclude traditional risk-limiting manual recounts done in conjunction with the cryptographic election if the ballots have a cryptographic overlay (as in Scantegrity II).

Blackbox assumption. Finally, like Punchscan, Scantegrity and Eperio, we do make a blackbox assumption that a perfectly private computation can be performed on a tamper-resistant device. Blackboxes are stateless devices without any non-volatile memory. The simply compute an output from a set of inputs without revealing any intermediary values in the function. They could be implemented as a hardware circuit, FPGA, or in software in a trusted execution environment such as Intel TXT (c.f., [30]).

Future work might explore the removal of this assumption, through a distributed computation, however we rely on it for this initial work in the area. We do note however that it is not immediately clear that a distributed computation is necessarily better. If an adversary wanted to attack the election by corrupting computational devices, it seems logical that compromising n devices is harder than compromising 1—in fact, this reasoning is seductive enough that the shareholders might use standard computers without extra precautions to perform their computations. In such case, compromising n devices might be as easy as compromising one (e.g., through an exploit for a common operating system) and might indeed be easier if the single blackbox device (it does not even need to be a full fledged computer) is given a lot of attention in terms of hardening it against attack.

Human-voteable and vote-and-go. Some voting schemes require the voter to participate in some multi-party computation. For example, [5] requires that voters take their vote and secret-share it with different election authorities. Even [29], a voting scheme "without cryptography," requires the voter to perform an amount of arithmetic which is arguably unreasonable in practice. In contrast, a human-voteable (also called barehanded [36]) voting scheme is one which does not require any kind of computational device to vote (such as a trusted computer).[1] Vote-and-go refers to the fact that individual voters are not expected to assist in any kind of post-ballot computations, such as computing the tally. All major governmental elections today are have both properties. It is difficult to see how a scheme that does not have both can escape being an impractical academic exercise. While smartphones are ubiquitous, their use opens up new attack vectors and is little better than trusting a polling machine or a physical ballot counted by humans.

4 Protocol Components

4.1 Verifiable Secret-Sharing and Commitment

A (k, n) verifiable secret-sharing (VSS) scheme is a multi-party protocol between a *dealer* and n *shareholders* that consists of two functions $\langle \mathsf{Share}, \mathsf{Recover} \rangle$. When invoking share, the dealer distributes some secret string x among the shareholders such that no subset of shareholders less than k can jointly output x and the dealer proves that each share can be consistently used to reconstruct some secret without an error. When invoking $\mathsf{Recover}$, k or more shareholders combine their shares to recover x (if less than k shareholders honestly contribute their shares, $\perp$ is recovered instead).

The guarantees of a VSS scheme can be made information-theoretic while tolerating up to $k < n/2$ malicious shareholders, assuming the existence of a broadcast channel. A broadcast channel is already a standard assumption in an E2E voting scheme. Many VSS schemes exist, each targeting different efficiency metrics. For our purposes, we assume the use of a standard scheme due to Rabin and Ben Or [35].

The relationship between a VSS scheme and a commitment function was explored recently by Garay et al. [20]. They observe that VSS is typically used a distributed 'analogue' to a commitment scheme and prove that VSS realizes a commitment-like properties. Informally speaking, the two main properties of bit-commitment are binding and hiding, which respectively mean that the sender can only open the commitment in one way, and that the receiver is unable to distinguish between (chosen) committed messages m_0 or m_1.

The respective properties of VSS which will act as the binding and hiding conditions are:

[1] Note we do not refer to assistive technology (AT) that helps voters with disabilities cast a vote—for this reason, we dislike the term barehanded. Rather we mean devices that are trusted to perform a computation for the voter, not navigate an interface.

- If no strict majority of shareholder's shares uniquely defines a secret, then there will be an abort. In other words, the dealer is unable to either create a commitment that they cannot open, or a commitment that can be opened in more than one way.
- No strict minority subset of shareholders can reconstruct the secret, or prevent an honest strict majority from reconstructing the secret. If a secret fails to be reconstructed, then the faulty shares can be identified. In other words, no strict minority subset of colluding sShareholders can change an existing commitment, or prevent the honest shareholders from opening the commitment.
- The secret will only be reconstructed when the majority of honest shareholders come to an agreement. In contrast to a two-party bit-commitment, the dealer is not involved in the opening process. Some pre-agreed condition will trigger the honest shareholders to divulge their shares. In our case, they are triggered by an auditor.

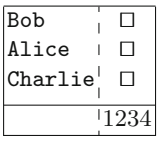
Concretely, given a (n,k)-VSS scheme, our commitment scheme will consist of two function $\langle \mathsf{Commit}, \mathsf{Open} \rangle$ realized as follows.

- $\mathsf{Commit}(x)$: The dealer takes a secret x and invokes $\mathsf{Share}(x)$ with the shareholders and proves that the shares are consistent. A failure of the secret-sharing is considered a failure of commitment. If successful, the dealer announces a commitment identifier id to the shareholders used to identify the commitment that should be opened. This identity is output as commitment value c (in a standard commitment, c would be functionally dependent on x).
- $\mathsf{Open}(c)$: The auditor sets $id = c$ broadcasts to the shareholders $\mathsf{Recover}(id)$. The honest shareholders follow the protocol to determine if the commitment should be opened or not. If so, they execute the reconstruct protocol and send to the auditor their shares, who reconstructs the secret. The honest majority will identify any dishonest shareholders, whose shares the auditors will ignore.

4.2 Eperio

Our voting protocol is based on the Eperio voting system [18]. Technically Eperio is a backend component that can realizes a variety of voting systems. We summarize some details of that protocol which we will augment with VSS in Sect. 5.

Ballots. Eperio can utilize different ballot types. We use a ballot in the style of Prêt à Voter (see Fig. 1): a permuted list of candidates with a serial number. The ballot is assumed to be physically unforgeable and is marked by the voter and split along the dotted line. The candidate ordering is shredded, while the mark position and serial number is optically scanned and then kept by the voter as a privacy-preserving receipt. In Fig. 1, we also show a tabular form of the ballot that is exactly equivalent. This form of the ballot could be printed out and given



U	M	S
1234.01	☐	Bob
1234.02	☐	Alice
1234.03	☐	Charlie

Fig. 1. A Prêt à Voter ballot with 3 candidates. Each ballot has a randomly shuffled order of candidates. Left side: the ballot as received by the voter. Right side: an equivalent formulation of the same ballot information in tabular form.

to voters, however it would be a poor design relative to the ballot form on the lefthand side of the figure.

The tabular form of the ballot consists of 3 columns and $\mathcal{C}$ rows, where $\mathcal{C}$ is the number of candidates in the election. The first column, which we denote by **U**, are Unique IDs which contains a unique ballot identifier and a choice identifier. In the example ballot of Fig. 1, the ballot number is 1234 and the suffixes identify each of the $\mathcal{C}$ markable positions on ballot 1234. So in this case, markable position 1234.01 would count for Bob. On a different ballot, say 1235, position 1235.01 might correspond to a different candidate.

The second column is the *Marks List* column, which we denote by **M**. In this column, the voter places a checkmark at exactly one spot, indicating the row corresponding to the candidate the voter wishes to vote for. The last column is the *Candidate Selection* column, which we will denote by **S**. This is a list of the candidates in a randomly permuted (per-ballot) order.

Eperio Tables. An Eperio table is a data structure that encodes the ballot information. If you were to take every ballot in tabular form, concatenate them end-to-end, you would end up with the 'canonical' Eperio table. This canonical table is never used directly, but many (e.g., 20) instances of it are created which are row-wise shuffles the table. In the original Eperio protocol, the **U** and **S** columns are individually encrypted for each instance of an Eperio table prior to the election to be used in the post-election audit.

Eperio Protocol. Prior to the election, a set of trustees use a blackbox device (trusted for ballot secrecy but not integrity) to generate a canonical Eperio table for an election with $\mathcal{C}$ candidates and $\mathcal{V}$ voters. All randomness used by the blackbox is deterministically derived from seeds provided by the trustees. The canonical table will be $3 \times \mathcal{CV}$. The canonical table is provided to the printers for printing the ballots. As in almost all paper-based E2E voting systems, printing is assumed to be a trustworthy process (at least with respect to ballot secrecy—a print audit will establish the correctness of the printed ballots but cannot distinguish between a malicious printer or honest printers being provide the wrong information to print).

A set of ℓ Eperio tables are generated by applying a random permutation to the rows of the canonical table by the blackbox. ℓ is a security parameter where

an attack that moves a vote from Alice to Bob will be detected (given adequate receipt checks and print audits) with probability $1 - 2^{-\ell}$. The **U** and **S** columns of each Eperio table is publicly committed prior to voting.

During voting, voters may request a ballot to be print audited (we defer to the paper the discussion of the print audit—we can handle more simply in our protocol). They then fill out their ballots for their selected candidates and have the mark position portion of their ballot recorded (they can keep this lefthand side of the ballot as a receipt). After the election, the trustees input into the blackbox their random seeds and the scanned ballots (**U** and **M**). The blackbox reconstructs all the tables and asserts an **M** column for each Eperio table. These **M** columns and an assertion of the final tally is published.

After the results have been asserted, a random beacon is used to select an ℓ-bit string; one bit for each Eperio table. If the bit for a given table is 0, the blackbox (again reseeded by the trustees) reveals the **U** column and if it is 1, it reveals the **S** column (the **M** column for each is already public). For each **UM**-revealed table, voters can check their receipt and everyone can check for consistency across each table. For each **MS**-revealed table, anyone can check that it matches the asserted tally. The specific reasoning for each of the three possible audits can be found in [18]. For any particular committed Eperio table, if only one of these combinations is opened, privacy is preserved.

5 Our Protocol

Our observation is that the encryption in Eperio is used as a commitment scheme and can be changed to any type of commitments. The authors themselves make this observation suggesting that the perfectly-binding commitment scheme (based on encryption) could be replaced with Pedersen commitments for everlasting privacy. We observe here that the commitments could be replaced with a VSS-style commitment to provide unconditional integrity and everlasting privacy (but sacrificing universal verifiability). Our protocol is given in Fig. 2.

Verification. In our protocol, voters may engage in three checks. The first is a receipt check, which applies to any tables opened **UM**. External auditors may also check with these tables that no ballot is over-voted. The second check is a print audit, which applies to all rows in each table corresponding to a print audited ballot opened **UMS**. The final check is the correctness of the tally, checked with **MS**. Note all **UM** tables are shuffled but otherwise identical versions of the same data, and likewise with all **MS** tables. The basic integrity attack a malicious blackbox can conduct is changing the tally, which constitutes moving marks in the **M**. However it must guess which tables will be opened **UM** and leave these unmodified (or the moved marks will be detectable via a receipt check), and guess exactly which tables will be opened **MS** to move the marks (or the tally will be unmodified, or inconsistent across tables). The probability of guessing correctly is $2^{-\ell}$ where ℓ is the number of tables. For $\ell = 20$ (a parameter used in Scantegrity for effectively the same purposes), the probability of guessing

Pre-Casting

1. Voters register with a local election authority. Issues of voter registration fraud are handled by the EA and are beyond the scope of this work.
2. The EA publishes the number of candidates $\mathcal{C}$ and number of ballots to print (e.g., $2 \cdot \mathcal{V}$ where $\mathcal{V}$ is the voting age population and the scalar 2 allows for, on expectation, one print audit per voter). The EA sets security parameter ℓ.
3. The blackbox uses local randomness to create the canonical Eperio table (which is provided to the printers) and ℓ permutations of it. It then uses VSS to commit the permuted tables to the shareholders, cell by cell. Each table's format and index is published. Upon completion, the shareholders purge the memory of the blackbox.

Vote Casting and Tallying

1. Voters show up and register at the designated voting locations. For each voter, the EA will give the voter a paper ballot, such as the one in Figure 1, assuming they have not voted already.
2. The voter may optionally choose to print audit the ballot. The scanner notes the serial number and its status as audited. The ballot is voided for voting purposes, and the voter is given the next ballot with the same option to audit or vote.
3. Once the voter decides to vote, she marks her ballot and destroys the portion of the ballot containing the candidate ordering. The other portion, containing the serial number and marked position, is copied by the scanner and the original is kept by the voter as a privacy-preserving receipt.
4. After the election, the scanners publish what they received: the **M** column of the canonical table.
5. A quorum of at least k honest shareholders submit their shares of all tables to the blackbox, which reconstructs the canonical table (by sorting each Eperio table and checking for consistency). It also takes as input the scanner data. It outputs an asserted **M** column for each of the ℓ tables and an asserted final tally. The shareholders publish the output and purge the blackbox's memory.

Audit

1. An unpredictable ℓ-bit value is publicly generated by a beacon (e.g., using stock prices [14]).
2. For bit i of the beacon value, a quorum of at least k honest shareholders publish their shares of each cell in the **U** column in the i-th Eperio table if the bit is 0, and each cell in the **S** column if the bit is 1. For print audited ballots (only), they publish both the **U** and **S** cells.
3. The shareholders securely delete all unused shares.

Fig. 2. Our variant of Eperio using VSS.

correct is less than a thousandth of a percentage. Importantly, this probability is independent of the adversary's computational power.

Discussion: Minimizing blackbox usage. The shareholders in our scheme are involved in three phases of the protocol: (1) preelection to use the blackbox to instantiate the election data, (2) after the election to use the blackbox to assert the mark column for each table, and (3) after the challenge to open up the data. In original Eperio, the blackbox must be used in all three steps. In our protocol, (3) can be accomplished by the shareholders directly without requiring the blackbox. In a variation of our protocol, we could also eliminate the blackbox from step (2). In step 2, the blackbox is required to permute a list of marks. The shareholders could do this directly if in step (1), the blackbox gave them each (in a specified order) a permutation to apply such that the composition of all these permutations is the permutation that was used. The issue is that this requres n-out-of-n shareholders in step (2) instead of k (however only k are required in step 3).[2]

6 Proof of Security Sketch

In our security proof sketch, we will reduce a breaking of either privacy or integrity to the breaking of one or more properties of the VSS scheme. We assume that the blackbox's computations are unobservable, and that the broadcast and private channels between shareholders are secure. In practice, these channels need not introduce extra cryptographic (and hence computational) assumptions, since they can be implemented as physical channels such as trusted couriers. In short, breaking either privacy or integrity will imply that strictly more than half of shareholders are malicious. Put differently, if a majority of shareholders collude (violating our assumptions), then they can determine how each voter voted (link ballot IDs to candidates voted for) and can modify the tally to anything they want and have it accepted by the verification step. If the blackbox assumption fails, the adversary can determine how each voter voted but cannot undetectably modify the tally.

6.1 Privacy

It was shown in Eperio [18] that violating privacy reduces to a number of assumptions including breaking the *hiding* property of the commitment. Since we effectively only change the commitment scheme, we can ask ourselves: "If a cabal of malicious shareholders, auditors and voters collude, can they break the hiding property of the VSS-commitment?" Assuming, as always, that the number of malicious shareholders is a strict minority, the answer to the above question is no.

We do not pursue a full simulation-based proof but we comment that VSS-commitments have an additional property that should streamline such a proof,

[2] Future work might explore the possibility of giving each shareholder a matrix that interpolates to the correct permutation matrix under the sequential composition of any k-out-of-n interpolations.

relative to the computational commitments used in Eperio. As a cut-and-choose protocol, Eperio faces a standard problem of simulateability: as the challenge space grows, the ability for the simulator to anticipate the correct challenge decreases exponentially (if it rewinds the verifier, it must do it an exponentially-increasing number of times which is not permissible). This can be side-stepped by, say, letting the simulator program the beacon value (by running it through a random oracle) or by repeating the protocol with one-bit challenges. In our case, a VSS-commitment is effectively a trapdoor commitment scheme for any majority of the shareholders. During the audit phase, the simulator can open a commitment in such a way that is perfectly consistent with any tally constraints imposed onto it.

Finally, we must also take care that each random choice (permutation in the tables) is truly random and not the result of a deterministic random generator (as in the original Eperio) or else the the permutations will not have a perfectly uniform distributed (which could be distinguished by an unbounded adversary). We modify Eperio along these lines—the shareholders do not contribute randomness, rather they remember shares of the randomness used (in the form of shuffled tables which can be resorted to recover the permutation).

6.2 Integrity

As in Eperio, the integrity of the election is reduced to a number of assumptions including the *binding* property of the commitment. We have replaced the commitment used by VSS, and in Sect. 4.1 we have argued that VSS has properties which corresponds to the binding property of a commitment scheme.

The auditing process remains the same. For each of the permuted Eperio tables, an auditor will ask the shareholders to open the commitments in such a way that corresponds to the three audits, as discussed in Sect. 4.2. Assuming that the number of malicious shareholders are strictly less than half, the VSS binding property guarantees that they cannot change the commitment that has been successfully executed.

In fact, let us suppose that the malicious shareholders can arbitrarily control where the marks go in the permuted Eperio tables. However, since there is at least one honest shareholder, the malicious shareholders do not know how to consistently mark the votes. Therefore, with high probability increasing exponentially to one in the number of Eperio tables, either a voter will detect that his vote is inconsistent with his receipt when the **U** columns are opened during the auditing process, or an auditor will discover inconsistencies across different Eperio tables opened the same way. In either way, the malicious shareholders' cheating is detected.

7 Conclusion

We present a system, based on Eperio, that offers integrity and ballot secrecy against computationally unbounded adversaries, regardless of whether such an

adversary is a voter, verifier, or election trustee. Further, our system enables voters to cast paper-based ballots, such as an optical scan ballot overlay as used in Scantegrity II or a permutation-style optical scan ballot as used in Prêt à Voter. Once the ballot is cast, the voter may leave and does not have to participate in tallying the election (in contrast to the other category of systems providing unconditional security: boardroom voting schemes).

To be even-handed, we point out that our system introduces several drawbacks. We rely on private and broadcast channels which, in practice, require computational cryptography, thereby negating information-theoretic security. We have argued that these channels may be implemented physically as untappable channels and in fact, for elections such as the Scantegrity II municipal election at Takoma Park, MD, election officials did meet in person in the same room to set-up the election and to compute the final tally. Like other paper ballot systems, the physical ballots are assumed to be unforgeable (therefore malicious voters cannot repudiate a correct audit) and we trust the EAs to not peek at the printed physical ballots before issuing them to voters (which would break privacy). Both of these issues could be mitigated to a large extent by using Scantegrity II ballots, however in Scantegrity II the scanner learns how the vote was cast (as it is a cryptographic overlay and not a replacement system).

Most importantly in terms of drawbacks, our system removes the ability for voters to independently verify the election results. They must trust that a majority of shareholders are honest. While we have no data on how many voters do a full cryptographic check of the election results in a typical E2E-verifiable election, we expect that many will already defer to someone else to check (whether by running their software without validating it or simply believing their assertions). That said, universal verification provides the agility to decide who you trust after the election and even do it yourself if you do not adequately trust anyone else who can perform the check. We are not advocating that unconditional security trumps universal verification, but we believe it is important to provide viable solutions for both sides of this trade-off. This way, readers can decide which is most appropriate for their election requirements.

Acknowledgements. We thank Claude Crépeau for helpful insights. We thank the anonymous reviewers who pointed out relevant work, suggested interesting ideas, and showed us where our paper needed more clarity. The second author acknowledges funding for this work from NSERC and FQRNT.

References

1. Adida, B.: Helios: web-based open-audit voting. In: USENIX Security (2008)
2. Bell, S., Benaloh, J., Byrne, M.D., Debeauvoir, D., Eakin, B., Kortum, P., McBurnett, N., Pereira, O., Stark, P.B., Wallach, D.S., Fisher, G., Montoya, J., Parker, M., Winn, M.: Star-vote: a secure, transparent, auditable, and reliable voting system. JETS **1**, 8 (2013)
3. Benaloh, J.: Simple verifiable elections. In: EVT (2006)
4. Cohen, J.D., Fisher, M.J.: A robust and verifiable cryptographically secure election scheme. In: SFCS (1985)

5. Broadbent, A., Tapp, A.: Information-theoretically secure voting without an honest majority. In: WOTE (2008)
6. Burton, C., Culnane, C., Schneider, S.: Verifiable electronic voting in practice: the use of vvote in the victorian state election. In: IEEE Security and Privacy (2016)
7. Carback, R.T., Chaum, D., Clark, J., Conway, J., Essex, A., Hernson, P.S., Mayberry, T., Popoveniuc, S., Rivest, R.L., Shen, E., Sherman, A.T., Vora, P.L.: Scantegrity II election at Takoma Park. In: USENIX Security Symposium (2010)
8. Chaum, D.: Elections with unconditionally-secret ballots and disruption equivalent to breaking RSA. In: Barstow, D., et al. (eds.) EUROCRYPT 1988. LNCS, vol. 330, pp. 177–182. Springer, Heidelberg (1988). https://doi.org/10.1007/3-540-45961-8_15
9. Chaum, D.: Secret-ballot receipts: true voter-verifiable elections. IEEE Secur. Priv. **2**(1), 38–47 (2004)
10. Chaum, D., Carback, R., Clark, J., Essex, A., Popoveniuc, S., Rivest, R.L., Ryan, P.Y.A., Shen, E., Sherman, A.T.: Scantegrity II: end-to-end verifiability for optical scan election systems using invisible ink confirmation codes. In: EVT (2008)
11. Chaum, D., Essex, A., Carback, R., Clark, J., Popoveniuc, S., Sherman, A.T., Vora, P.: scantegrity: end-to-end voter verifiable optical-scan voting. IEEE Secur. Priv. **6**(3), 40–46 (2008)
12. Chaum, D., Ryan, P.Y.A., Schneider, S.: A practical voter-verifiable election scheme. In: di Vimercati, S.C., Syverson, P., Gollmann, D. (eds.) ESORICS 2005. LNCS, vol. 3679, pp. 118–139. Springer, Heidelberg (2005). https://doi.org/10.1007/11555827_8
13. Chevallier-Mames, B., Fouque, P.-A., Pointcheval, D., Stern, J., Traoré, J.: On some incompatible properties of voting schemes. In: Chaum, D., Jakobsson, M., Rivest, R.L., Ryan, P.Y.A., Benaloh, J., Kutylowski, M., Adida, B. (eds.) Towards Trustworthy Elections. LNCS, vol. 6000, pp. 191–199. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-12980-3_11
14. Clark, J., Hengartner, U.: On the use of financial data as a random beacon. In: EVT/WOTE (2010)
15. Cramer, R., Franklin, M., Schoenmakers, B., Yung, M.: Multi-authority secret-ballot elections with linear work. In: Maurer, U. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 72–83. Springer, Heidelberg (1996). https://doi.org/10.1007/3-540-68339-9_7
16. 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). https://doi.org/10.1007/3-540-69053-0_9
17. Demirel, D., van de Graaf, J., dos Santos Araujo, R.S.: Improving Helios with everlasting privacy towards the public. In: EVT/WOTE (2012)
18. Essex, A., Clark, J., Hengartner, U., Adams, C.: Eperio: mitigating technical complexity in cryptographic election verification. In: EVT/WOTE (2010)
19. Gallegos-Garcia, G., Iovino, V., Rial, A., Ronne, P.B., Ryan, P.Y.A.: (Universal) unconditional verifiability in e-voting without trusted parties. Technical report, IACR Eprint Report 2016/975 (2016)
20. Garay, J., Givens, C., Ostrovsky, R., Raykov, P.: Broadcast (and round) efficient verifiable secret sharing. In: ICITS (2014)

21. Goldwasser, S., Kalaj, Y.: On the (in)security of the Fiat-Shamir paradigm. In: FOCS (2003)
22. Hao, F., Zieliński, P.: A 2-round anonymous veto protocol. In: Christianson, B., Crispo, B., Malcolm, J.A., Roe, M. (eds.) Security Protocols 2006. LNCS, vol. 5087, pp. 202–211. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-04904-0_28
23. Henry, K., Stinson, D.R., Sui, J.: The effectiveness of receipt-based attacks on threeballot. IEEE TIFS **4**(4), 699–707 (2009)
24. Hosp, B., Vora, P.L.: An information-theoretic model of voting systems. Math. Comput. Model. **48**, 1628–1645 (2008)
25. Kiayias, A., Yung, M.: Self-tallying elections and perfect ballot secrecy. In: Naccache, D., Paillier, P. (eds.) PKC 2002. LNCS, vol. 2274, pp. 141–158. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45664-3_10
26. Kiayias, A., Zacharias, T., Zhang, B.: End-to-end verifiable elections in the standard model. Technical report 2015/346, IACR Eprint Report (2015)
27. Locher, P., Haenni, R.: Verifiable internet elections with everlasting privacy and minimal trust. In: Haenni, R., Koenig, R.E., Wikström, D. (eds.) VOTELID 2015. LNCS, vol. 9269, pp. 74–91. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-22270-7_5
28. Locher, P., Haenni, R., Koenig, R.E.: Coercion-resistant internet voting with everlasting privacy. In: Clark, J., Meiklejohn, S., Ryan, P.Y.A., Wallach, D., Brenner, M., Rohloff, K. (eds.) FC 2016. LNCS, vol. 9604, pp. 161–175. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53357-4_11
29. Malkhi, D., Margo, O., Pavlov, E.: E-voting without 'Cryptography'. In: Blaze, M. (ed.) FC 2002. LNCS, vol. 2357, pp. 1–15. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-36504-4_1
30. Mannan, M., Kim, B.H., Ganjali, A., Lie, D.: Unicorn: two-factor attestation for data security. In: CCS (2011)
31. Moran, T., Naor, M.: Receipt-free universally-verifiable voting with everlasting privacy. In: CRYPTO (2006)
32. Moran, T., Naor, M.: Split-ballot voting: everlasting privacy with distributed trust. In: CCS (2007)
33. Neff, C.A.: A verifiable secret shuffle and its application to e-voting. In: CCS (2001)
34. Popoveniuc, S., Hosp, B.: An introduction to punchscan. In: WOTE (2006)
35. Rabin, T., Ben-Or, M.: Verifiable secret sharing and multiparty protocols with honest majority. In: Proceedings of the Twenty-first Annual ACM Symposium on Theory of Computing, STOC 1989, New York, NY, USA, pp. 73–85. ACM (1989)
36. Riva, B., Ta-Shma, A.: Bare-handed electronic voting with pre-processing. In: Proceedings of the USENIX Workshop on Accurate Electronic Voting Technology, EVT 2007, Berkeley, CA, USA, pp. 15–15. USENIX Association (2007)
37. Rivest, R.L., Smith, W.D.: Three voting protocols: threeballot, VAV, and twin. In: EVT (2007)
38. Schoenmakers, B.: A simple publicly verifiable secret sharing scheme and its application to electronic voting. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 148–164. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48405-1_10
39. Schoenmakers, B.: Fully auditable electronic secret-ballot elections. Xootic Mag. **8**, 5 (2000)

40. Stadler, M.: Publicly verifiable secret sharing. In: Maurer, U. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 190–199. Springer, Heidelberg (1996). https://doi.org/10.1007/3-540-68339-9_17
41. 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). https://doi.org/10.1007/978-3-642-38980-1_28