# Securing Optical-Scan Voting

Stefan Popoveniuc[1], Jeremy Clark[2], Richard Carback[3],
Aleks Essex[4], and David Chaum[5]

[1] George Washington University
[2] University of Waterloo
[3] University of Maryland, Baltimore County
[4] University of Ottawa
[5] Votegrity

**Abstract.** This paper presents a method for adding end-to-end verifiability to any optical-scan vote counting system. A serial number and set of letters, paired with every candidate, are printed on each optical-scan ballot. The letter printed next to the candidate(s) chosen by the voter is posted to a bulletin board, and these letters are used as input to Punchscan's verifiable tallying method. The letters do not reveal which candidate was chosen by the voter. The method can be used as an independent verification mechanism that provides assurance that each vote is included in the final tally unmodified—a property not guaranteed by a manual recount. We also provide a proof-of-concept process that allows the election authority to settle disputes after the polls close while preserving ballot secrecy.

**Keywords:** anonymity, cryptography, E2E, mix networks, optical-scan, privacy, Punchscan, security, universal verifiability, voting.

## 1 Introduction

Abraham Lincoln once observed that democracy is "the government of the people, by the people and for the people." The foundation of any government *by the people* rests on a society's ability to hold inclusive elections and accurately count every vote. Unfortunately, the introduction of new voting technology in some countries, including the United States, has diminished voters' confidence in the security of their democratic contribution.

At the time of writing, the most prominent voting technology used in US elections are optical-scan systems [EDS06]. These systems provide two methods for counting votes. Under precinct scanning[1], the scanned ballots are electronically tallied at the precinct after the polls close. Tallies produced this way rest on software security, and recent security reviews of certified optical-scan systems have demonstrated serious vulnerabilities that undermine the trustworthiness of tallying through this method [WJB06, KMRS06, ea07]. The second tallying

[1] Alternatively the ballots can be scanned centrally, however this requires greater reliance on chain-of-custody.

D. Chaum et al. (Eds.): Towards Trustworthy Elections, LNCS 6000, pp. 357–369, 2010.

method is to conduct a manual recount. Hand counting ballots is slow and prone to human error, but most seriously, manual recounts do not detect if ballots were replaced or modified.

This paper[2] proposes a simple way to add a third, superior vote counting method to any optical-scan voting system—a method that is *end-to-end (E2E)* verifiable. Our method does not interfere with the mechanics of the optical-scan procedures. It permits voters to mark the ballot exactly as in the underlying system, candidates can be listed in a fixed order, the electronic tally can be performed on the same equipment, and manual recounts can be conducted as necessary. Our method only requires the central election authority to print additional information on each ballot and follow some additional procedures after the polls close. Procedures at the precinct remain essentially unchanged. Thus, we introduce no additional risk of equipment failure and only generate a marginal increase in cost. In return, each voter gains the ability to check her choices are included in the tally unmodified and everyone can check that all included ballots are counted correctly.

Our method evolved from the Punchscan voting system [PH06], which contains two key elements: (1) The front-end which is the ballot, how it is constructed, marked, scanned, and posted; and (2) the back-end which provides a universally verifiable method for recovering voter choices from the posted information without compromising ballot secrecy. We introduce significant modifications to the front-end by pairing symbols with candidate names on each ballot. The symbol paired with each chosen candidate is posted on a bulletin board. The back-end only changes semantically by decoding what each letter means in a verifiable way instead of decoding a position marked on the ballot. In addition, we provide a dispute resolution process which can be conducted after the polls close.

This paper is organized as follows. Preliminaries are provide in the next section. Section 3 outlines our motivation for creating this method and our design goals for the new ballot style which is introduced in Section 4. The voter experience at the precinct is explained in Section 5, while section 6 gives an overview of how the tally is generated using the Punchscan back-end. Section 7 presents a dispute resolution process for proclaimed discrepancies between a voter's receipt and the bulletin board. Finally, Section 8 provides implementation details and Section 9 contains our concluding remarks.

## 2 Related Work

End-to-End (E2E) voting systems (also known as receipt-based or universally-verifiable voting systems) represent a class of systems that offer unconditional integrity of election results, usually by using cryptographic techniques. They include VoteHere's MarkPledge [Nef04a, Nef04b], Punchscan [ECCP07], Prêt

[2] This paper is derived from a presentation by the first author at *Frontiers of Electronic Voting* in 2007. Conceptually, the ideas herein predate a more thorough treatise of the system to appear in *IEEE Security and Privacy*.

à Voter [CRS04], Scratch & Vote [AR06] and Voter Initiated Auditing [Ben07]. Typically, an E2E system will provide the voter with a signed or stamped receipt of her vote. To preserve ballot secrecy, the receipt does not reveal how she voted but contains an indirect representation of her vote—either through encryption or a permutation-based obfuscation. The polling place equipment records this same indirect representation and never sees the voter's actual choices. After the polls close, the election authority publishes all of the representations it received on a public bulletin board, allowing voters to check that their choices are included and unmodified. In the case of an error, the signature on the receipt provides the voter with proof of a discrepancy—an event that can trigger a variety of responses, depending on election policy.

The election officials will generate the final tally by recovering the votes from the indirect representations of the ballots, either through decryption or inverting the obfuscating permutation. To preserve voter privacy, the tally is generated through a specially designed protocol that explicitly removes the correspondence between a receipt and the choices it represents. To ensure the unconditional integrity of this protocol, a mandatory auditing process is performed on the protocol which proves to a mathematical certainty that all ballots were recovered properly and that the choices were unmodified—whether by a software error, a malicious election official, or a hacker. This auditing process can be independently duplicated by anyone.

## 3 Design Goals

The front-end of Punchscan has been criticized for its use of indirection, which could introduce voter intention errors and significantly increase time-to-vote. Filling out a Punchscan ballot requires the voter to find a letter corresponding to the candidate of choice in a row of randomly ordered symbols and then marking the corresponding hole. To date, no peer-reviewed user study has been performed on the usability of a Punchscan ballot and so these criticisms are unsubstantiated, however they appear reasonable.

A trade-off involved with the use of Punchscan is that it cannot produce on-demand ballots. This is because it requires paper with holes punched in unique positions for each election. While drilling holes through paper is very fast and cheap, it does require special equipment. The system is also environmentally wasteful, requiring the destruction of one of the two paper sheets composing the ballot.

Another issue is that custom software to scan a Punchscan ballots is needed. While current optical-scan voting equipment could be used to acquire an image of the ballot, the hardware was developed to perform mark sense scanning, which detects whether a shape has been filled in or an arrow has been drawn. The software for scanning Punchscan ballots is simple but the cost and inconvenience of upgrading the software on already-owned optical scanners is a serious impediment to the adoption of Punchscan.

Punchscan uses an unfamiliar mechanism for counting votes. While it provides exceptionally high integrity, it does not comply with legislation and voting

standards that require hand-countable paper ballots for manual recounts. A preference for elements that voters and election officials are used to could ultimately weigh the decision to adopt new voting technology away from systems like Punchscan. Having only minimal changes that can be easily explained and understood by those running the elections is critical for the success of the adoption process.

With regards to privacy, Punchscan ballots are better than optical-scan ballots. With Punchscan, the scanner does not get to see the full ballot, only the receipt which does not indicate how the voter voted. Additionally, any fingerprints that are left by the voter on the scanned paper are irrelevant. However this makes it impossible to have results reported by the precinct optical scan or to conduct a manual recount. Even though Punchscan ultimately offers a stronger proof of accuracy than even a manual recount, this is a legal impediment to Punchscan's adoption in many jurisdictions.

Taking these criticisms, trade-offs, and shortcomings into consideration, we have produced a set of design goals for a new ballot style that can interface with Punchscan's back-end. This is not meant as a replacement for Punchscan, nor should it be thought of as an improvement in all regards. Some jurisdictions may opted for the improved privacy properties offered by Punchscan *in lieu* of the ability to conduct manual recounts or use existing equipment. Because the back-end is shared with Punchscan, ballots could be mixed and matched in the same election. Some ballots can be printed in Punchscan style, while others can be of the new type. To summarize, our design goals are as follows:

1. Eliminate indirection,
2. Allow on-demand printing,
3. Use a single sheet,
4. Use a familiar method for marking the ballots,
5. Allow the use of existing voting equipment without upgrades,
6. Do not interfere with optical-scan tallying,
7. Do not preclude the option of a manual recount,
8. Allow Punchscan's privacy-enhanced ballots to be used in conjunction.

## 4 Ballot Design

As will become evident, one major advantage of our technique is the ability to preserve the ballot layout that is imposed by the law or an equipment manufacturer. With our method, the order in which the candidates appear on the ballot can be the same on each ballot. We illustrate a typical optical scan ballot configuration in Figure 1. Our method adds two elements to this ballot: symbols that are paired with each candidate and a serial number represented in a form that a mark sense scanner can read.

Before the election, a set of symbols is published for every contest on the ballot. These symbols can be letters, numbers, shapes, or multi-character codes. A fixed *canonical order* of the symbols, corresponding to candidate order, is also established. The order itself is arbitrary and we use alphabetical order from

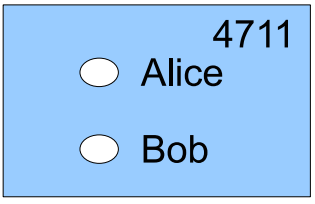


**Fig. 1. Normal Ballot.** A typical optical scan ballot configuration. Candidates are listed in a fixed order across all ballots and there is a designated location next to each candidate that a voter marks.

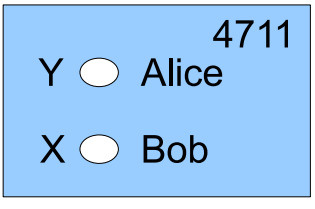


**Fig. 2. New Ballot Configuration.** Next to each oval there is a different symbol.

here onward. For each ballot, the symbols are shuffled relative to the *canonical order* and printed next to the ovals associated with the candidates as illustrated in Figure 2). Figure 3 illustrates the set of possible symbol orders for a two-candidate contest. Note that on any two different ballots, the symbols associated with the same candidate may be different. Therefore if the distribution of possible symbol orders is uniformly random across the ballots, the knowledge that a vote was cast for any particular symbol provides one with no statistical advantage in determining which candidate was voted for.

The serial number of each ballot is printed in such a way as to allow easy scanning. Assuming that scanner uses the mark sense technology, the serial number is represented using a matrix of digits, where the digits of the serial number blackened out. This is illustrated in Figure 4. Using this representation, any existing mark sense scanner can be configured to recognize a serial number even if its ignorant of the fact its scanning a serial number as opposed to votes. Mark sense attempts to determine if the geometric shape is filled in or not. Sets of shapes can be defined such that only one shape is allowed to be filled in, as would be done in for each contest on a first-past-the-post ballot, and if it more than one appears to be filled, the darkest is typically selected. The output of the scanner is an *electronic ballot images (EBIs)* which is a list of each shape and its state—filled or unfilled. For our method, the information stored by the precinct scanners will be the positions that were filled in, where some of these positions represent the serial number.

The last aspect of the new ballot is the reserved a portion that is to be used as a stub—detachable along a perforation in the paper. The stub also bares the serial number of the ballot printed on it but this serial number is for the voter to

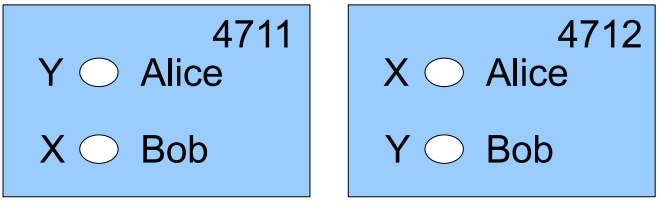


**Fig. 3. Different Ballots.** On two different ballots, the order of the symbols associated with the candidates may be different.

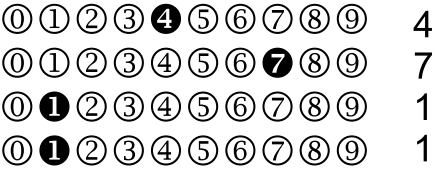


**Fig. 4. Representing the serial number.** This format allows a mark sense scanner to read it by checking which digits are black.

read, not the scanner, and can be represented in numeric form. A ballot adapted from a real election is shown in Figure 5. We now address the purpose of this stub and how it is used by the voters.

## 5 The Voting Ceremony

On voting day, after proper identification and verification, the voter is issued our modified ballot by the poll workers. She fills out this ballot as she would the ballot of any optical-scan system, filling in the ovals beside the candidates she wishes to vote for. Before she scans the ballot, she detaches the ballot stub and may write on it or any piece of paper the symbols that corresponded to the candidates she chose. Afterward, the ballot is scanned and placed into a ballot box as under the normal procedures of optical-scan voting.

While in the booth, if the voter makes a mistake while filling in her ballot, the ballot is officially marked as spoiled and given back to the voter. Also, the voter may choose to deliberately spoil a ballot in addition to the ballot she votes on. If she wishes to do so, she must tell the election official before getting her ballot. The election official will then present the voter with two ballots, both faced down, such that the voter cannot see the order of the symbols. The voter may choose one of the ballots to spoil and the other one to vote on. In section 6, we describe how these spoiled ballots can be used to audit the integrity of the election. The voter may keep the spoiled ballot for herself or she may choose to give them to an organization that she trusts (*e.g.* The League of Women Voters) to use in checking the integrity of the election on her behalf.

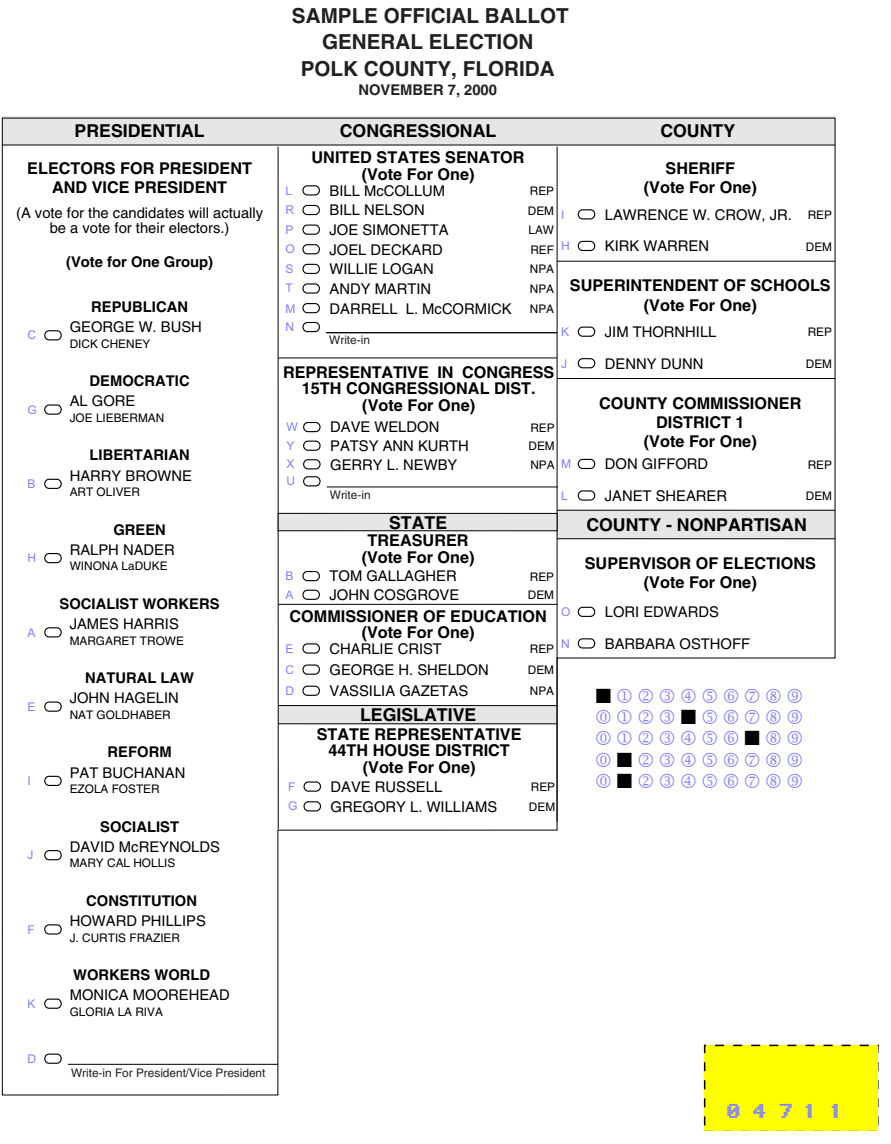


**Fig. 5. Ballot adapted from Florida's 2000 Polk County Election[pol07].** Attached to a ballot there is a stub with the serial number of the ballot printed on it.

After the polls close, anyone can go to the *bulletin board*—an official election web site—and enter the serial number of the ballot from a ballot stub. The *bulletin board* responds with a list of symbols corresponding to the symbols that were paired with the chosen candidates on the ballot.

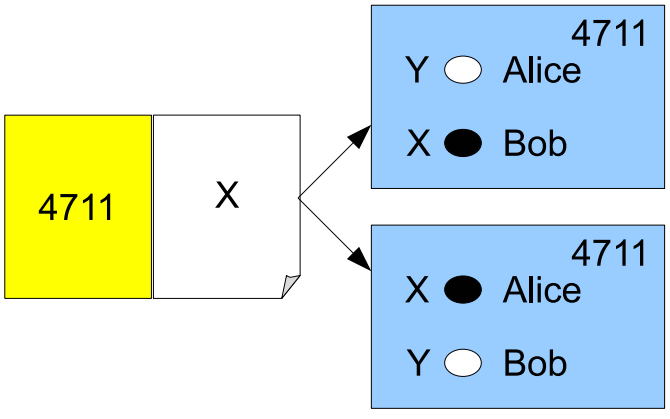


**Fig. 6. Plausible Deniability.** Knowing which symbol was marked does not reveal which candidate was chosen, since on different ballots the same symbol can correspond to different candidates.

Since the same symbol corresponds to different candidates on different ballots, the *bulletin board* preserves ballot secrecy while providing enough information to verifiably calculate election results. For example, if a coercer attempts to influence a voter to cast a vote for "Alice" and sees on the bulletin board that the ballot with her serial number had the oval corresponding to symbol "X" marked, symbol "X" may correspond to "Alice" or "Bob" as shown in Figure 6. Thus, the coercer has no assurance that the voter followed his directions. Even if she wanted to, the voter cannot provide any evidence that any symbol corresponds to a certain candidate for a ballot, since the paper she used to write down the symbols is merely a personal record and does not bear any value for anyone else but the voter.

If, after checking the *bulletin board*, the voter sees the same symbols she recorded on her piece of paper, she is guaranteed the following three things:

1. The scanner properly read her ballot.
2. The scanner correctly posted the symbol.
3. The central election authority received and counted a ballot with the correct symbol.

More generally, the voter is given assurance that her vote was properly interpreted and received by the election authority. However she still does not know if her ballot was counted correctly for candidate she chose. The next section presents the mechanism that ensures everyone, not only voters, that all the votes were counted as recorded—*i.e.,* that for each given serial number, the symbols that are posted on the bulletin board are counted as votes for the same candidate that the symbols appeared beside on that ballot.

## 6 Generation of the Election Results

In the previous sections, we have described a new front-end for the Punchscan voting system. We give a brief, conceptual overview of Punchscan's back-end, and then address how it is used to produce a universally verifiable set of election results.

### 6.1 Punchscan's Back-End: The Punchboard

The back-end of the Punchscan system, which translates voter marks into voter choices, is called the Punchboard, and it is a specially constructed anonymity network. A variety of anonymity networks have been proposed in literature, most of which are based on Chaum's mix networks [Cha81]. In a mix network, anonymous messages are encrypted multiple times (forming an 'onion') and then sent through a series of nodes, each of which remove the order-based correspondence between their input and output message set by using a secret permutation, and remove content-based correspondence by decrypting each message once. A second type of anonymity network, also due to Chaum, is the DC-net [Cha88] which hides an anonymous message amidst random numbers that cancel out when summed together, revealing the message. The Punchboard is both distinct from these two anonymity networks and similar in certain regards.

Similar to a mix network, the Punchboard operates on a batch of input messages (voter marks) and produces an output set (voter choices) in a permuted order. However unlike a mix network, the Punchboard does not distribute trust among multiple nodes. Instead the permutations are produced from a secret election-wide key that is shared among multiple trustees in a threshold scheme. This election key is never stored in memory but is regenerated by a cryptographic combination of trustee-supplied pass phrases on each occasion that the Punchboard is needed. The election trustees can be comprised of adversarial opponents, such as representatives of each political party, along with government officials to ensure the group has no incentive to collude.

Before the election commences, the election trustees use this election-wide key to generate the secret permutations in the Punchboard. A cryptographic commitment for each is computed and published. These commitments will later form the basis of a proof that the Punchboard has not been altered through the course of the election. This proof can be verified by any independent, interested party. There is one path per ballot for the election. An advantage of the Punchboard over the use of a mix network, which are used in some other voting systems [CRS04], is that the Punchboard does not use encryption and decryption functions. Rather it uses a much faster method based on modular addition. The voter marks can be thought of as the sum of some random numbers (represented on a Punchscan ballot by the random order of the symbols) and the position marked by the voter. The first node of the Punchboard adds an additional random number to this sum so it is untraceable if one were to examine the input and output set of the first node. The second node subtracts off the sum of all the random numbers leaving the position of the candidate to

receive a particular ballot's vote[3]. As with the secret shuffles, all the random numbers to be used are generated from the election-wide key and fixed prior to the election through a cryptographic commitment.

Assuming that randomized partial checking [JJR02] is used for auditing the Punchboard, a minimum of two nodes are sufficient. The commitments to the random numbers can be independent, or can be blended into the commitments to the paths, meaning that a path and a set of numbers are committed to at the same time, using a single commitment. The Punchboard is very fast because it can be implemented using an efficient hash-based commitment scheme and without the use of public key or symmetric key cryptography. Full details of the Punchboard have been omitted from this document but are available in previous publications [PH06, ECCP07].

### 6.2 Using the Punchboard with the New Ballots

To ensure that the ballots are printed properly, some printed ballots are checked in an audit. This audit, which is similar to the audit in [AN06], is performed by a *designated challenger*, who is a voter or representative of an independent organization at the poll site. The designated challenger chooses a ballot from the set of ballots. A poll worker then records the ballot as spoiled, marking all the contests on it so that it cannot be used in the election, and gives it to the designated challenger. For each spoiled ballot, the election authority reveals the entire path through the Punchboard. The designated challenger is then able to check the revealed path against what was committed to for that ballot, and be assured that each symbol paired with the proper candidate and therefore would have been properly counted. This process can be repeated until a predefined statistical certainty is reached that all the ballots are printed exactly as they were committed to.

In Punchscan, top layer symbols and bottom layer symbols are permuted from the canonical order according to two statistically independent permutations. In the new ballot style, the two permutations are composed such that marking a certain symbol on the new ballot is equivalent to marking a certain hole in Punchscan—*e.g.* marking the second symbol on the new ballot is equivalent to marking the second hole on a Punchscan ballot in terms of which candidate receives the vote. Using this method, the voter marks on the new ballot can be transformed into voter choices using a Punchboard identical to the ones used for Punchscan ballots. Indeed, this actually permits the new ballots and Punchscan ballots to be mixed within the same election, and the ballot styles can be made indistinguishable in their presentation on the bulletin board.

Since the optical scanner cannot read the symbols that were marked, it reads the candidates that were chosen on each ballot. After receiving the electronic ballot images from the polling place scanners, the election authority transforms these clear votes into marked symbols. The information required to perform this transformation is stored on the Punchboard. After this step, the symbols and the results are posted on the official web site.

[3] Instead of numbers in $Z_n$ and modular addition, permutations and permutation composition can be used, or more general any elements in a group and the group operation.

## 7 Dispute Resolution

Any *challenger* in possession of a ballot stub, usually a voter, may initiate the dispute resolution process if she believes the record on the *bulletin board* is incorrect. This process provides a privacy preserving method to resolve discrepancies between voter records and the *bulletin board*. The reason records may not be the same are usually caused by two situations:

1. **The Voter is Wrong.** Because there is no control over what symbol the voter records, she could have recorded a symbol other than the symbol paired with her chosen candidate.
2. **The Record is Wrong.** A scanner or software error may cause the recorded vote to be incorrect. It is also possible that an attacker may have changed or altered ballots, or that some other malfeasance occurred.

In 1, dispute resolution convinces all *challengers* that each ballot was received, unaltered, and counted correctly. In 2, dispute resolution provides proof to any observer that the record on the *bulletin board* is correct. In both situations, dispute resolutions preserves ballot privacy.

The dispute resolution protocol is carried out between the election *official* and a set of *challengers* as follows:

1. **Proving the Ballot is Present and Unaltered**
   (a) *Challengers* present ballot stubs.
   (b) *Official* retrieves ballots with serial number from each stub and places them into a privacy sleeve that does not reveal candidate choices, but does show the rest of the ballot including the part of the ballot where the stub was taken and the back of the ballot. It should also show candidate lists, but should hide the choices and symbols.
   (c) *Challengers* verify ballot sheet has not been modified and may conduct forensic analysis to verify the ballot stub was at one point attached to the ballot presented by the *official*.
2. **Showing the Selected Ballot Letters**
   (a) For each ballot with the same letter marked, the *Official* moves the ballot to a separate privacy sleeve being careful to hide the choices made on the ballot (e.g. with the back of the ballot facing the *challengers*). This sleeve does not show the serial number but does show the choices and symbols of the race in dispute. *Official* then drops each ballot into an empty lottery-style hopper[4].
   (b) All ballots with the same symbol marked are mixed, and removed from the hopper.
   (c) Each *challenger* verifies that all ballots have the same symbol next to the chosen candidate.
   (d) This process repeats for all sets of ballots with different symbols marked.

[4] If not enough ballots are available, the *official* can add fake ballots to the hopper.

After all disputes are settled, everyone can assume that the public record of chosen symbols is correct, and that no ballots were lost. If necessary, officials re-compute the results from the corrected public record.

The current method may be unable to ballot secrecy if the race in question is too long or not enough ballots with the same symbol marked are challenged. Finding a more efficient dispute resolution procedure that does not require physical interaction or forensic analysis is an open problem to be addressed in future work, as are the best methods for dealing with ballots with write-ins.

## 8 Implementation

Given that the speed of the system is largely dependent on the speed of the Punchboard, the performance of our method is nearly equivalent to that of Punchscan. However, we have produced an implementation and tested with both moderate and large-sized elections. On a 1.73 GHz laptop, we were able to tabulate 1 million ballots in under 10 minutes. Using actual statistics from Florida's 2000 Polk County election as a benchmark [pol07], where there were 32 contests with an average of 3.2 candidates per contest, we tabulated 200,000 ballots in under 4 minutes and audited the Punchboard in less than 2 minutes.

The complete source code written in Java, as well as object code, for our implementation is available [web07], along with instructions on how to build it and use it in a mock election. The object code is directly accessible from any browser, via Java Network Launching Protocol (JNLP), without installing any of the cryptographic libraries our implementation depends or performing any other specific configuration.

Future work in this area includes developing an easier and more efficient dispute resolution process or minimizing its need by using other techniques, procedures for limiting access to the ballots before and after they are handed out to the voters, addressing forced randomization attacks and simplifying the Punchboard for the sake of efficiency and explanatory ease, especially with individuals unfamiliar with verifiable mix networks or other anonymity networks.

## 9 Concluding Remarks

In this paper, we have created a new front-end for Punchscan with numerous improvements. The new ballots eliminate the indirection in order to increase usability. They are printed on a single sheet, which allows more efficient use of resources, and the sheets do not contain holes which allows for on-demand printing. The ballots are filled out exactly as in optical-scan voting which should be a method familiar to many voters in the US. However, the greatest advantage of our method is that it can be used as an add-on to the optical-voting systems already owned by many election districts without software upgrades. Our method does not interfere with the tally produced by optical-scan equipment, nor does it preclude the option of a manual recount. Without detracting in any way, we add end-to-end verifiability to a popular voting system and provide unconditional assurance that every vote was counted accurately.

## References


[AN06] Adida, B., Andrew Neff, C.: Ballot casting assurance. In: EVT 2006: Proceedings of the USENIX/Accurate Electronic Voting Technology Workshop 2006 on Electronic Voting Technology Workshop, Berkeley, CA, USA, p. 7. USENIX Association (2006)

[AR06] Adida, B., Rivest, R.L.: Scratch & vote: self-contained paper-based cryptographic voting. In: WPES 2006: Proceedings of the 5th ACM workshop on Privacy in electronic society, pp. 29–40. ACM Press, New York (2006)

[Ben07] Benaloh, J.: Ballot casting assurance via voter-initiated poll station auditing. In: Preproceedings of the 2007 USENIX/ACCURATE Electronic Voting Technology Workshop (EVT 2007) (August 2007)

[CEC+08] Chaum, D., Essex, A., Carback, R., Clark, J., Popoveniuc, S., Sherman, A., Vora, P.: Scantegrity: End-to-end voter-verifiable optical-scan voting. IEEE Security & Privacy 6(3), 40–46 (2008)

[Cha81] Chaum, D.L.: Untraceable electronic mail, return address, and digital pseudonym. Communication of ACM (February 1981)

[Cha88] Chaum, D.: The dining cryptographers problem: unconditional sender and recipient untraceability. J. Cryptol. 1(1), 65–75 (1988)

[CRS04] Chaum, D., Ryan, P.Y.A., Schneider, S.A.: A Practical, Voter-verifiable, Election Scheme. Technical Report Series CS-TR-880, University of Newcastle Upon Tyne, School of Computer Science (December 2004)

[ea07] Bowen, D., et al.: California secretary of state, voting systems review: Top-to-bottom review (2007), `http://www.sos.ca.gov/elections/elections_vsr.htm`

[ECCP07] Essex, A., Clark, J., Carback, R.T., Popoveniuc, S.: The Punchscan voting system: VoComp competition submission. In: Proceedings of the First University Voting Systems Competition (VoComp) (2007)

[EDS06] Election Data Services. 2006 voting equipment study (October 2006)

[JJR02] Jakobsson, M., Juels, A., Rivest, R.L.: Making mix nets robust for electronic voting by randomized partial checking. In: Proceedings of the 11th USENIX Security Symposium, Berkeley, CA, USA, pp. 339–353. USENIX Association (2002)

[KMRS06] Kiayias, A., Michel, L., Russell, A., Shvartsman, A.A.: Security assessment of the diebold optical scan voting terminal (October 2006)

[Nef04a] Andrew Neff, C.: Practical high certainty intent verification for encrypted votes (October 2004)

[Nef04b] Andrew Neff, C.: Verifiable mixing (shuffling) of el-gamal pairs (October 2004)

[PH06] Popoveniuc, S., Hosp, B.: An introduction to PunchScan. In: IAVoSS Workshop On Trustworthy Elections (WOTE 2006), Robinson College, Cambridge UK (June 2006); Also GWU-CS Technical Report

[pol07] Election Website for Polk County, Florida (October 2007), `http://www.polkelections.com/`

[web07] Punchscan voting system website (October 2007), `http://www.punchscan.org/`

[WJB06] Wagner, D., Jefferson, D., Bishop, M.: Security analysis of the diebold accubasic interpreter (February 2006)