Single Layer Optical-scan Voting with Fully Distributed Trust*

Aleksander Essex[1], Christian Henrich[2], and Urs Hengartner[1]

[1] Cheriton School of Computer Science
University of Waterloo
Waterloo, ON, Canada N2L 2G1
{aessex,uhengart}@cs.uwaterloo.ca
[2] Institut für Kryptographie and Sicherheit/EISS
Kahrlsruhe Institute of Technology
76128 Karlsruhe, Germany
{christian.henrich}@kit.edu

Abstract. We present a new approach for cryptographic end-to-end verifiable optical-scan voting. Ours is the first that does not rely on a single point of trust to protect ballot secrecy while *simultaneously* offering a conventional single layer ballot form and unencrypted paper trail. We present two systems following this approach. The first system uses ballots with randomized confirmation codes and a physical in-person dispute resolution procedure. The second system improves upon the first by offering an informational dispute resolution procedure and a public paper audit trail through the use of self-blanking invisible ink confirmation codes. We then present a security analysis of the improved system.

1 Introduction

Research into cryptographically "end-to-end" verifiable *optical-scan* voting systems has come a long way toward practicality. This progress has not come easily: academics and election administrators often struggle to agree on a vast and often orthogonal set of core system properties. Similar in spirit to Benaloh [2], we advocate the *coexistence* of modern cryptographic proofs of correctness and conventional, lower-tech, methods for auditing elections. In this paper we tackle a long standing trade-off of properties in the voting literature: distributed trust versus a conventional optical-scan paper ballot form.

Typically cryptographic voting schemes allow the voter to construct a receipt of their vote enabling each voter to confirm the inclusion of their ballot in the election tally. In order to protect ballot secrecy, the association between a receipt and the corresponding (clear-text) vote must be kept hidden at all times. Many proposals have relied on trusted entities or hardware to enforce this, especially with regards to ballot printing. Other proposals distribute trust among multiple entities through the use of specialized multi layer ballot forms.

* Full version available: http://eprint.iacr.org/2011/568

Our Proposal We consider a list of requirements for end-to-end verifiable optical scan voting that factors a diverse set of stakeholders (i.e., cryptographers, election officials, legislators, democracy groups, etc). This list is by no means exhaustive and does not encompass challenges faced by other voting methods (e.g., internet, mail-in, etc). Our list is as follows:

1. **Distributed trust**: No single party, *including* the ballot printer(s), gains an advantage in deducing how a voter voted or in linking a receipt to its corresponding clear-text vote. This is a vital requirement of any secret ballot election employing the receipt paradigm.
2. **Single layer ballot form**: A ballot is a single sheet of paper with a *fixed order* candidate list[3] and the voter marks the optical scan ovals *directly beside* their chosen candidate. Multi layer ballots are an artifact of cryptographic voting, requiring voters to re-learn how to cast a ballot. Our experience in running real-world cryptographic elections—both with single layer and with multi layer ballot forms—has indicated to us that multi layer ballots are more cumbersome for voters and more difficult to administer for election officials [14, 5, 41].
3. **Human-readable paper audit trail**: Pursuant to the legal requirements of many jurisdictions voting, voting intent remains plainly evident on cast ballot forms. Such an audit trail also allows for recoverability in the event of lost or forgotten cryptographic keys or other unforeseen errors.
4. **Public paper audit trail**: The collection of cast ballot forms (i.e., the *paper audit trail*) can be made public without revealing the link between receipt and clear-text vote. A public audit paper trail may also be a legal requirement and is critical in protecting ballot secrecy during a manual recount.

In this paper we propose two novel end-to-end verifiable optical scan voting systems that meet all four of these requirements. Some of these properties have been examined in the literature, but no proposal has achieved all of them. Scantegrity achieves 2 and 3 [9, 7]. Prêt à Voter and Scratch & Vote achieve 2 and 4 [10, 37, 1, 42], two Punchscan variants achieve only 4 [19, 22], and each of Split-Ballot Voting, ClearVote and Kusters *et al.* achieve 1 and 4 [28, 34, 23]. A proposal due to Benaloh [2] achieves 2, 3, and 4. See the section on related work for additional discussion.

Contributions We present two novel systems for single layer optical-scan voting with distributed trust based respectively on the ballot styles used by Scantegrity [9] and Scantegrity II [7].

Basic System: We propose a basic two-party system for creating ballot forms with randomized confirmation codes that meets properties 1, 2, and 3. It

[3] There are also potential advantages to using ballots with randomized candidate lists. Our system can accommodate this approach with minor protocol changes.

relies on a private paper audit trail and an in-person physical dispute-resolution procedure.

Improved System: We then propose an improved two-party system that uses 'self-blanking' invisible ink confirmation codes. It improves on the basic system by allowing the paper audit trail to be made public, thereby achieving all four properties. In addition it offers an *informational* dispute-resolution procedure allowing disputes to be resolved based on knowledge of a confirmation code (as opposed to physical possession of a receipt).

2 Preliminaries

2.1 Physical Primitives

End-to-end verifiable ballots often employ physical security methods as part of the receipt creation process. The use of physical security mechanisms can be contentious due to inherent questions regarding their cost, feasibility, and real-world security properties. However, there is precedent for protocols built around *ideal* physical security mechanisms (c.f. [18, 27]). Throughout the rest of this paper we assume that all physical security mechanisms function ideally. Broadly speaking the ballot secrecy properties of our systems reduce to those of Scantegrity's when the physical security mechanism fail. A brief discussion of this is included in our security analysis in the full paper. [4]

Physical Security Mechanisms We briefly summarize the physical security mechanisms employed by our systems.

Invisible ink as its name implies is initially invisible when printed and becomes visible only after *activation*. It was proposed for use in the Scantegrity II system [7], and has been implemented and fielded in a live municipal election in the United States [5]. For the improved system presented in Section 4 we additionally make use of a 'slow' developing ink,

Scratch-off coating is a convenient, cost-effective and widely available method for concealing (and subsequently revealing) printed information. It has been employed in several voting schemes (cf. [1, 39]) to protect ballot secrecy,

Visual cryptography [29] is a well known technique for visually implementing a logical exclusive disjunction (i.e., an *xor*) built from a physical medium acting as a logical disjunction (i.e., an *or*). A message or graphical image can be split into two or more information-theoretically secure shares. When the shares are combined (i.e., overlayed) the message becomes visually perceptible.

[4] http://eprint.iacr.org/2011/568

Physical Security Sub-protocols We briefly summarize the physical security sub-protocols used by our systems.

Document Authenticity: We require a method for determining a document's authenticity. Classical methods for anti-counterfeiting (e.g., watermarks, holographic foil, embedded magnetic strips, etc) can be cost-prohibitive. Paper fibre analysis (cf. [12]) using commercial-grade scanners is possible[5]. For the sake of our description we assume that there exists an efficient physical scheme for determining a ballot's authenticity,

Private Printing: we make use of private printing techniques to pick and print *human-readable* confirmation codes on ballots without either printer individually knowing which codes were printed. A proposal for two-party private printing was made in [15]. Private printing is used in the improved system.

2.2 Cryptographic Primitives

We briefly outline the main cryptographic primitives used by our systems. We note that these primitives are standard across the cryptographic voting literature.

Homomorphic Encryption Let $\langle \mathsf{DKG}, \mathsf{Enc}, \mathsf{DDec} \rangle$ be a *distributed public-key encryption* scheme. Without loss of generality, DKG generates two private key shares x_1 and x_2 for parties $\mathcal{P}_1$ and $\mathcal{P}_2$ respectively and a joint public key Y. Encryption $[\![m]\!] = \mathsf{Enc}_Y(m, r)$ is semantically secure and homomorphic in at least one operation. Decryption $m = \mathsf{DDec}_{(x_1, x_2)}([\![m]\!])$ requires both key shares. Specifically we will make use of exponential Elgamal [13] with distributed decryption [33]. For simplicity we will omit the public-key when implied. We additionally require a *partially-homomorphic xor* operation $\tilde{\oplus}$ such that, for a pair of messages $m_1, m_2 \in \{0, 1\}$, $[\![m_1]\!] \tilde{\oplus} m_2$ produces a ciphertext that encrypts the bitwise xor of the associated plaintext bits, i.e., $[\![m_1 \oplus m_2]\!]$. We present a bit encryption scheme based on exponential Elgamal in the full paper[4] though there is more than one way to accomplish this (cf. [20, 30]).

Mixnets Mixnets have long been a fixture in cryptographic voting. We make use of a simple re-encryption mixnet (cf. [31]) structure to create our proofs (we do not utilize a separate proof of correct mixing, as it is provided by other parts of our system). *Re-randomization* (a.k.a., re-encryption) of a ciphertext c is accomplished by computing $c' = \mathsf{ReRand}(c, r) = c \cdot \mathsf{Enc}(0, r)$[6]. By rerandomizing and shuffling a batch of ciphertexts we implement a simple *reencryption mixnet*, Mix. In this paper, when applying Mix to a matrix of ciphertexts, we describe mixing as occurring on *tuples* of ciphertexts grouped by columns and shuffled by rows.

[5] In general there are privacy threats due to fingerprinting documents however this is not a threat to ballot secrecy assuming non-collusion.

[6] Replace 0 with the *identity* element for other groups.

Commitments We use a cryptographic *commitment* scheme to commit to permutations as part of a cut-and-choose proof of shuffle. The dispute resolution procedure in the improved system requires the prover to either unveil (i.e., *de*-commit to) the code, or alternatively to issue a *non-interactive proof of plaintext inequality*. A commitment inherent to IND-CPA secure encryption fits this dual role. Here a sender *commits* to a message m by posting its encryption $[\![m]\!] = \mathsf{Enc}(m, r)$. Later the commitment can be unveiled when the sender reveals an m', r', allowing anyone to verify $\mathsf{Enc}(m', r') = [\![m]\!]$, and hence $m' = m$. This approach is commonly used in several voting schemes (e.g., [3, 1, 40]).

Non-interactive Challenges As part of our cut-and-choose correctness proof we require a method for fairly generating random challenge bits. Loosely speaking, *fairness*, requires that no one is able to predict, or controllably influence the output with non-negligible advantage. Furthermore, the fairness of the method should be *convincing* to voters. Both the heuristic due to Fiat and Shamir [17], and the notion of a *random beacon* (cf. [36, 11]) are possibilities.

2.3 Participants

There are several entities that participate in the election.

- A set of **voters** with the authority to cast a ballot in the election, optionally construct a privacy-preserving receipt of their vote, and optionally participate in an election audit,
- An **election operations commission** $\mathcal{C}$ with the capability and authority to organize and run an election, operate a polling place, optically scan ballots, report results, act as a custodian of the cast ballot record, and participate in an in-person dispute resolution procedure,
- Two independent **ballot printers** $\mathcal{P}_1, \mathcal{P}_2$ who possess the capability and authority to print documents in the untrusted printing model and participate in a secure (cryptographic) two-party computation,
- An election **scrutineer** $\mathcal{S}$ with the authority to audit the correctness of printed ballots relative to their cryptographic representation. Additionally $\mathcal{S}$ acts as a proxy for voters during disputes with $\mathcal{C}$ to protect their identity. In practice there might be any number of election auditors, representing the candidates or other democracy groups.

As a fundamental requirement of our security model, we assume that neither printer nor election commission collude with one another.

3 The Basic System

The basic system produces a public and universally verifiable cryptographic proof attesting to the correctness of the election's outcome. This correctness proof is

based on standard cut-and-choose techniques (cf. [9, 7, 8]). Without loss of generality we consider a single-contest election involving n ballots[7] and m candidates. The basic system involves several protocols. The protocols generateBallots, preElectionPrep, postElectionPrep encompass the preparation for the public election audits. Note that each of these protocols taken individually is only secure in an *honest-but-curious* setting. To make them robust against an active adversary we make use of a set of *audit* protocols proveScan, proveReceipt, provePrinting and resolveDispute. A summary of notations used is presented in Table 1.

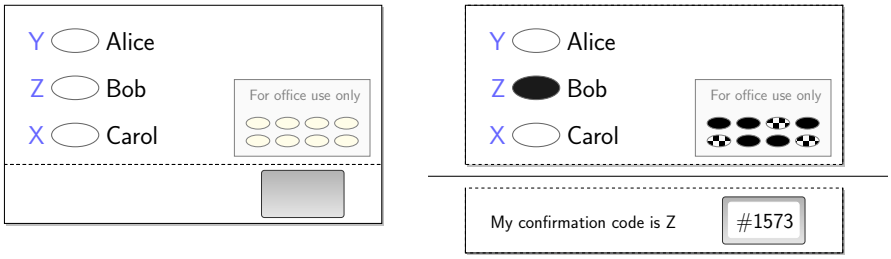
n	Number of ballots to print	T	List of all ballot-tuples
m	Number of candidates	BallotTable	Table of ballot information
d	Bit-length of ballot-id	ReceiptTable	Table of receipt information
L	List of candidate names	$\mathsf{MP}_1/\mathsf{MP}_2$	Printer 1/2's master permutation
Σ	Confirmation code alphabet	π/ρ	Random perm'ns composing to MP_1
α	Soundness parameter	σ/τ	Random perm'ns composing to MP_2
b/B	Ballot-id/list of ...	MidMarks	Intermediate mark state list
r/R	Receipt-id/list of ...	MidMarksP1	$\mathcal{P}_1$'s intermediate mark state list
c/C	Confirmation code/list of ...	MidMarksP2	$\mathcal{P}_2$'s intermediate mark state list
μ	Mark-state of opscan oval	eid	Election-unique identifier

Table 1: Notations

The Ballot The basic optical-scan paper ballot form has a pre-printed, fixed-order candidate list $L = \{l_1 \dots l_m\}$. Adjacent to each candidate is an optical scan oval with a *mark state* $\mu \in \{0, 1\}$ corresponding respectively to whether the oval was unmarked or marked. The ballot form is separated into two regions by a perforation. The top constitutes the *ballot portion*, and the bottom is the *receipt portion*. An alphabet Σ of m confirmation codes is defined. Each optical scan oval (and hence each candidate) is associated with a confirmation code drawn independently at random, and without replacement, from Σ. A *ballot-id* b is a d-bit[8] vector printed on the ballot portion. An independent *receipt-id* r is printed on the receipt portion. The first printer prints the receipt-ids under a scratch-off coating and the second prints the confirmation codes. Both printers will jointly print the ballot-id in invisible ink. Printing of the ballot- and receipt-ids is done such that each printer only knows what *it* prints (and not what its counterpart prints). The basic ballot is depicted in Figure 1(a).

[7] The number of ballots printed is the total number of voters times a *heuristically* chosen expansion factor to account for audited and spoiled ballots.

[8] Since in the basic scheme ballot-ids are the xor of random bit vectors, d is chosen to be large enough so as to make duplicate ballot-ids highly unlikely.



(a) Unmarked ballot form.

(b) **Top:** Cast ballot portion with activated ballot-id. **Bottom:** Completed receipt portion with revealed receipt-id.

Fig. 1: **Basic ballot**: Optical-scan ballot form with *ballot portion* (top) and tear-off *receipt portion* (bottom) depicting a randomized confirmation code list, a unique ballot-id printed in *invisible ink visual-crypto* and a unique receipt-id beneath a scratch-off coating. Ballot printing is distributed between two printers such that neither can match receipts with cast ballots.

Ballot tuple A ballot is fully specified by the tuple $\{b, r, c\}$, which denotes the association between a unique *ballot-id* bit vector $b \in \{0,1\}^d$, a unique *receipt-id* $r \in \{1 \dots n\}$, and a random permutation of *confirmation codes* $c = \pi(\Sigma)$ for a permutation π drawn independently and uniformly at random from the set of possible permutations of Σ.

3.1 Election Preparation

The election is initialized as follows: election commission $\mathcal{C}$ initializes a *public bulletin board* $\mathcal{BB}$[9] and a unique election identifier *eid*. Printers $\mathcal{P}_1$ and $\mathcal{P}_2$ jointly run DKG. They post the public key Y to public bulletin board $\mathcal{BB}$ and retain their respective private key shares x_1, x_2. This list of public parameters $\mathsf{pubParam} = \{n, m, d, L, \Sigma, \alpha, eid, Y\}$ is posted to $\mathcal{BB}$. All functions/protocols accept $\mathsf{pubParam}$ as input.

Ballot Tuple Creation The printers now jointly generate encrypted ballot tuples by running $\mathsf{generateBallots}$. This protocol is given in Algorithm 1.

Ballot Printing The n ballot forms are printed in three steps. For each ballot-tuple a paper ballot is prepared in the following order:

- **Static background:** directions, candidate names, etc, printed in black ink,
- $\mathcal{P}_1$**'s share:** the receipt-id is printed and concealed under scratch-off coating, $\mathcal{P}_1$'s share of the ballot-id printed in invisible ink visual-crypto,

[9] Typically modelled as an append-only broadcast channel with state (cf. [4]).

Algorithm 1: generateBallots

Participants: Printers $\mathcal{P}_1, \mathcal{P}_2$

Printer $\mathcal{P}_1$ should:
  for $i \in \{1 \dots n\}$ **do**
    Encrypt vectors of random bits:
    $B'(i) \leftarrow (\mathsf{Enc}(\mathsf{randBit}), \dots, \mathsf{Enc}(\mathsf{randBit}))$
    Post a non-malleable commitment to each randBit along with the random factor used to encrypt it.
  Encrypt and shuffle receipt-ids:
  $R \leftarrow \mathsf{Shuffle}(\mathsf{Enc}(1) \dots \mathsf{Enc}(n))$
end
Printer $\mathcal{P}_2$ should:
  for $i \in \{1 \dots n\}$ **do**
    Randomly shuffle and encrypt code confirmation codes:
    $C(i) \leftarrow \mathsf{Shuffle}(\mathsf{Enc}(\Sigma(1)) \dots \mathsf{Enc}(\Sigma(m)))$
end
Both Printers should:
  Simultaneously and respectively output B', R and C to $\mathcal{BB}$.
end
Printer $\mathcal{P}_2$ should:
  for $i \in \{1 \dots n\}$; $j \in \{1 \dots d\}$ **do**
    Homomorphically xor random bits:
    $b'_1 \dots b'_d \leftarrow B'(i)$
    $B(i) \leftarrow (b'_1 \tilde{\oplus}\ \mathsf{randBit}, \dots, b'_d \tilde{\oplus}\ \mathsf{randBit})$
    Post a non-malleable commitment to each randBit along with the random factor used in computing the xor.
  Output B to $\mathcal{P}_1$
end

//*Remark:* $\mathsf{Shuffle}(X)$ *applies a permutation to a list* X*, drawn independently and uniformly randomly from the set of permutations of size* $|X|$*.* randBit *returns a single bit drawn independently and uniformly at random. It is possible that* $\mathcal{P}_2$ *might attempt to maliciously select its bits as a function of* $\mathcal{P}_1$*'s. However* $\mathcal{P}_2$ *will not know (beyond a guess) what to print on the ballot, and will be caught in* ProvePrinting *with statistical certainty.*

– **$\mathcal{P}_1$'s share:** the confirmation codes are printed in regular ink, $\mathcal{P}_2$'s share of the ballot-id printed in invisible ink visual-crypto over $\mathcal{P}_1$'s share.

The completed ballot forms are then randomly shuffled and delivered into the custody of the election commission $\mathcal{C}$. Throughout the ballot printing and voting phases the printers will conduct random audits of ballot forms to ensure their authenticity and to look for signs of tampering (e.g, to catch if someone reveals the secret information then replaces the ballot with a replica). Note that if either printer prints something *other* than their contribution in generateBallots (e.g., if

a printer prints an all-black VC pixel), this will be caught in provePrinting with statistical confidence dependent on the number of audited ballots.

Pre-Election Proof Preparation The printers initialize the public audit dataset and cut-and-choose correctness proofs by running preElectionPrep. This protocol is given in Algorithm 2.

Voting and Receipt Creation An individual wishing to vote shall attend the polling place and authenticate themselves to $\mathcal{C}$. All qualified and authenticated individuals (i.e., voters) are then eligible to receive a ballot. The voter selects a ballot form at random from a stack of unmarked ballot forms and takes it, a regular (black) marking pen, and a privacy sleeve into a private voting booth. The voter marks the oval next to their preferred candidate l_i on the ballot portion. Then, if they so choose, the voter creates a receipt of their vote by noting the code letter c_i and writes it in the appropriate space on the receipt portion. The voter then places the marked ballot form into the privacy sleeve and returns it to the poll worker. The poll worker confirms the receipt-id's scratch-off coating is still intact and the ballot-id has not been activated (rejecting the ballot in such a case), then detaches the receipt portion and places it on a table in view of the voter. The ballot portion is then fed into the optical scanner. If the ballot is accepted the receipt portion is retained by the poll worker. If the ballot portion is successfully cast, the receipt portion is returned to the voter and the voting process is complete. A diagram showing completed ballot and receipt portions is depicted in Figure 1(b).

A Note about Timing Attacks In some jurisdictions, poll workers keep a poll book of voter identities in the *order they voted*. If the scanner were to likewise maintain the order of cast ballots it, taken along with the poll book, would compromise ballot secrecy. Since in our case the ballot is drawn at random from the pile, and the poll worker does not see the ballot- or receipt-ids, this threat can be mitigated by having voters cast ballots into a ballot box at the polling place and then scanning them later at a central location.

Post-Election Proof Preparation After the election $\mathcal{C}$ populates the BallotTable with the mark state information collected by the optical scanners. With this data the printers and can now finalize the cut-and-choose correctness proof by running postElectionPrep. This protocol is given in Algorithm 3.

3.2 Audits

There are three simultaneous properties that must be proven in order for the overall results to be proven correct. These audits include,

- **Proving correct mark-state reporting by** $\mathcal{C}$: Using their receipt, a voter $\mathcal{V}$ checks whether $\mathcal{C}$ correctly registered their vote by running proveScan,

Algorithm 2: preElectionPrep

Participants: Printers $\mathcal{P}_1, \mathcal{P}_2$
Public Input: Candidate list L
Private Input: Lists of encrypted ballot-ids B, receipt-ids R, and code shuffles C

Both Printers should:
  //Expand the n ballot tuples into a table of mn rows (one for every candidate on every ballot):
  for $i \in \{0 \dots n-1\}$ **do**
    $c_1 \dots c_m \leftarrow C(i)$
    for $0 \leq j \leq m-1$ **do**
      $T(1, mi+j) \leftarrow B(i)$
      $T(2, mi+j) \leftarrow \mathsf{Enc}(L(j+1))$
      $T(3, mi+j) \leftarrow R(i)$
      $T(4, mi+j) \leftarrow c_j$
  //$\mathcal{P}_1$ followed by $\mathcal{P}_2$ using master permutations MP1 *and* MP2 *respectively:*
  $T' \leftarrow \mathsf{Mix}(T)$
  //Create ballot and receipt tables:
  $\mathsf{BallotTable} \leftarrow \mathsf{DDec}(T'(1 \dots 2, :))$
  $\mathsf{ReceiptTable} \leftarrow \mathsf{DDec}(\mathsf{Mix}(T'(3 \dots 4, :))$
  Post $\mathsf{BallotTable}, \mathsf{ReceiptTable}$ to $\mathcal{BB}$
end
//Prepare cut-and-choose proof of correspondence between elements in the ballot and receipt tables:
Printer $\mathcal{P}_1$ should:
  for $i \in \{1 \dots \alpha\}$ **do**
    Choose $\pi_i \in_R \Pi_{mn}$
    Set ρ_i such that $\rho_i \circ \pi_i = \mathsf{MP}_1$
    Post $\mathsf{Commit}(\pi_i), \mathsf{Commit}(\rho_i)$ to $\mathcal{BB}$
end
Printer $\mathcal{P}_2$ should:
  for $i \in \{1 \dots \alpha\}$ **do**
    Choose $\sigma_i \in_R \Pi_{mn}$
    Set τ_i such that $\tau_i \circ \sigma_i = \mathsf{MP}_2$
    Post $\mathsf{Commit}(\sigma_i), \mathsf{Commit}(\tau_i)$ to $\mathcal{BB}$
end
//Remark: Let $x \in_r \Pi_y$ denote a permutation function x drawn independently and uniformly at random from the set of permutations of list of y elements. Let $\mathsf{MP}_1, \mathsf{MP}_2 \in_R \Pi_{mn}$. Then for $i \in \{1 \dots \alpha\}$, we have $\tau_i \circ \sigma_i \circ \rho_i \circ \pi_i = \mathsf{MP}_2 \circ \mathsf{MP}_1$.

Algorithm 3: postElectionPrep

Participants: Election Commission $\mathcal{C}$, Printers $\mathcal{P}_1, \mathcal{P}_2$
Private Input: Secret Master permutations $\mathsf{MP}_1, \mathsf{MP}_2$, Scanned Cast Ballots

//Populate BallotTable *with scanner data*
Election commission $\mathcal{C}$ should:
  foreach $\{b, s, \mu\}$ *recorded by scanner* **do**
    Find i for which $\mathsf{ballotTable}(1, i) = b$
    and $\mathsf{ballotTable}(2, i) = s$
    $\mathsf{ballotTable}(3, i) \leftarrow \mu$
  Post $\mathsf{ballotTable}(3, :)$ to $\mathcal{BB}$.
end
//Propagate marks from BallotTable *to* ReceiptTable
Printer $\mathcal{P}_1$ should:
  $\mathsf{MidMarks} \leftarrow \mathsf{MP}_1(\mathsf{BallotTable}(3, :))$
  Post MidMarks to $\mathcal{BB}$ **for** $i \in \{1 \dots \alpha\}$ **do**
    $\mathsf{MidMarksP1}_i \leftarrow \pi_i(\mathsf{BallotTable}(3, :))$
    Post $\mathsf{MidMarksP1}_i$ to $\mathcal{BB}$
end
Printer $\mathcal{P}_2$ should:
  $\mathsf{ReceiptTable}(3, :) \leftarrow \mathsf{MP}_2(\mathsf{MidMarks})$
  Post $\mathsf{ReceiptTable}(3, :)$ to $\mathcal{BB}$. **for** $i \in \{1 \dots \alpha\}$ **do**
    $\mathsf{MidMarksP2}_i \leftarrow \sigma_i(\mathsf{MidMarks})$
    Post $\mathsf{MidMarksP2}_i$ to $\mathcal{BB}$
end

- **Proving mark-state propagation by $\mathcal{P}_1, \mathcal{P}_2$**: The printers prove to any interested party that they honestly applied their master permutations to mark state information in BallotTable by running proveReceipt,
- **Proving printed ballot forms match $\mathcal{BB}$**: A scrutineer $\mathcal{S}$[10] runs provePrinting with the printers to verify that the ballot tuple information conveyed by the paper ballot forms *matches* the ballot tuple representation in $\mathcal{BB}$. Audited ballots are *spoiled* and not counted.

Because receipt creation is unsupervised, a dispute may arise between $\mathcal{C}$ and $\mathcal{V}$ over the correct confirmation code. In such an event a dispute resolution protocol can be run. For space reasons we defer complete listings of proveScan, proveReceipt, provePrinting and the dispute resolution procedure to the full paper.[4]

[10] A scrutineer is not strictly necessary. Voters themselves may choose to initiate this audit, although in our experience they rarely do!

4 Improved System

In this section we present a system that improves upon the basic system in two ways: First, it replaces the physical dispute resolution procedure with an *informational* dispute procedure. Second, the collection of cast ballots (i.e., the paper audit trail) can be viewed publicly without compromising ballot secrecy.

Informational Dispute Resolution The dispute resolution procedure of the basic system is inefficient and time consuming. Chaum et al. proposed the notion of invisible ink confirmation codes in Scantegrity II [7] as an *informational* means of resolving dispute. Under this approach, codes are printed in invisible ink, and only revealed to the voter if marked. Assuming the code space is sufficiently large so as to make successful random guess unlikely, then knowledge of *any* valid code can be taken as evidence that a voter correctly created their receipt. Any discrepancy found between a receipt and the ReceiptTable can then be attributed to $\mathcal{C}$ (assuming the other correctness proofs are valid). In the improved system, we create and print the codes using a *private printing* protocol. Thus the role of invisible ink is twofold: it restricts the voter's knowledge of unmarked codes *and* it prevents the printers from linking receipts to votes.

Public Paper Trail Invisible ink confirmation codes require a code space that makes random guessing statistically unlikely. For example Scantegrity II proposes a 3-digit code (making a random guess successful 0.1% of the time on average). However in the presence of unique (or semi-unique) codes, access to cast ballots coupled with the public audit dataset is sufficient (or nearly sufficient) to allow *any* observer to link receipts to clear-text votes. This not only means that the paper ballot record must be kept *secret*, but further that the custodian of the ballot record (i.e., $\mathcal{C}$) is trusted with knowledge of how voters voted. This is one of the major limitations of Scantegrity II. To address this privacy weak-spot, we require a method for not only privately printing a confirmation code, but for displaying it *only while the voter is in the booth*. In the presence of "disappearing" codes, not only can we offer distributed trust with respect to $\mathcal{P}_1, \mathcal{P}_2$ and $\mathcal{C}$, but we can also make the paper ballot record public.

Self-blanking Confirmation Codes We propose a method for printing of confirmation codes that is self-blanking (i.e., the message is only temporarily visible). The standard invisible ink described by Scantegrity II activates *instantaneously*. That is to say, the chemical reaction responsible for the ink's pigmentation completes on the order of milliseconds. It was suggested in [7] that a *slower* reacting ink might by the addition of an *anti-catalyst*. This substance, if present, can slow down pigmentation by seconds or minutes (depending on design needs). Combining the technique of visual cryptography with such a 'slow' invisible ink, we can construct a self-blanking pixel (see Table 2). Finally, combining self-blanking pixels with the private printing protocol of [15], we can print confirmation codes that are both distributed between two-parties and self-blanking.

a	b	$\mathsf{VC}(a)$	$\mathsf{VC}(b)$	Result when activated $t = 0$	$t > 0$	$t >> 0$
0	0	■∅	∅■	▒▒	▓▓	■■
0	1	■∅	■∅	▒■	▓■	■■
1	0	∅■	∅■	■▒	■▓	■■
1	1	∅■	■∅	▒▒	▓▓	■■

Table 2: **Self-blanking VC Pixel**. Two sub-pixels contain invisible ink. Each party applies an anti-catalyst (cyan) to *one* sub-pixel. Sub-pixels containing this substance darken more slowly than those without ($t = 0$ is the moment of activation). Eventually all sub-pixels darken "blanking" the pixel's value.

The Improved Ballot The improved ballot differs from the basic ballot in that it makes use of *self-blanking invisible ink* confirmation codes. The codes are printed inside the optical scan ovals in *self-blanking invisible ink*. When the voter marks an oval using the specially provided activator pen, the confirmation code is revealed allowing the voter (finite) opportunity to write down the code on their receipt. Eventually the oval darkens completely indicating *that* the oval was chosen by the voter, but not what the confirmation code was (see Figure 2).

(a) $t = 0$ (b) $t > 0$ (c) $t >> 0$

Fig. 2: **Optical-scan oval with self-blanking confirmation code** after being marked with an activator pen ($t = 0$ is the moment of activation).

Changes to the protocols The addition of self-blanking invisible-ink confirmation codes induces some changes the protocols presented in Section 3. Details are presented in the full paper[4] and are summarized as follows:

- **Ballot tuples**: $\mathcal{P}_2$ generates ballot-ids. Both printers run a *private printing* protocol to select a confirmation code and distribute it to VC shares,
- **Ballot printing**: $\mathcal{P}_2$ prints ballot-ids in invisible ink. Both printers print their shares of the confirmation codes using self-blanking visual crypto pixels,
- **Informational dispute resolution**: As in Scantegrity II, the printers only publish the confirmation code corresponding to the voted candidate. In the

case of a dispute, the printers jointly issue a non-interactive proof of plaintext inequality between all remaining (unencrypted) codes on the disputed ballot.

5 Security Analysis of the Improved System

For space reasons we defer our security analysis to the full paper.[4] To briefly summarize our results, owing to the similarities between systems, we reduce the correctness of the improved system to that of Scantegrity II. Although Scantegrity has been peer reviewed and used in a real election we are not aware of a formal proof of the correctness. A proof of correctness of Eperio, a related system, does offer some insight into how such a proof would proceed [16]. With respect to secrecy we present an argument that the improved system protects voter privacy even when one printer is corrupted. Assumptions regarding the physical primitives can be found there as well.

6 Related Work

We review some work related to verifiable voting systems with optical-scan paper ballots. This literature can be roughly separated into two categories: systems using single layer ballot forms but reliant on trusted parties/hardware and systems with distributed trust but with multi layer ballot forms.

Single layer ballot forms with trusted components The Scantegrity [9] and Scantegrity II [7, 8, 5] systems offer both a simple single layer fixed candidate list and an unencrypted paper trail, but make extensive use of trusted components to protect ballot secrecy including a computer for blackbox construction of the correctness proofs, the polling place scanner, the ballot printer as well as the custodian of cast ballots. Additionally the paper record reveals the link between receipt and clear-text vote making it unsuitable for public viewing. The Prêt-à-Voter [10, 37] system and its variants [1, 42, 38] also offer the voter a single-layer ballot form with randomized candidate list. Although the correctness proofs are usually described as a multi-party computation, ballot forms are generated by a trusted printer. Cast ballots are generally "encrypted" though variants exist that leave a human readable paper trail [26, 16]. Benaloh [2] proposes that receipts be generated and printed by a special-purpose device connected to the optical scanner. This has the distinct advantage that the ballots contain no identifying information (beyond the vote). However the issue of trusted ballot printing instead becomes a matter of trusted receipt printing.

Distributed trust with multi layer ballot forms Kubiak [22] and Carback et al. [19] propose *mostly* distributed modifications of the Punchscan system [35]. The former still relies on a trusted ballot printer, the later distributes printing but still relies on trusted hardware to generate ballot tuples. Carback and Popoveniuc [34] later propose a *three*-party distributed version of Punchscan in

which top- middle- and bottom-sheet permutations are each generated by independent printing authorities. In all cases voters must use an indirect marking procedure. Moran and Naor [28] propose an improved multi layer ballot form that does not rely on indirection and with considerably stronger, provable, security properties. Voters are issued layers in separate sealed envelopes. Once inside the booth the voters are directed to remove each layers from its envelope and stack the layers in a particular order. The resultant candidate list is horizontally offset from the optical scan ovals by a randomized amount. Lundin et al. [25] propose a distributed construction of the Prêt-à-Voter ballot based on a form of dealerless 2-party visual cryptography. The voter must be careful to align the VC shares in the booth in order to reconstruct the candidate list. Most recently Küsters et al. [23] present a version of Prêt-à-Voter system without a trusted printer, physically implementing a re-encryption mixnet using scratch-off coatings. The voter receives a separate ballot for *each* candidate, which can be cumbersome for races involving more than a few candidates.

Other schemes Chaum proposed the first physical receipt based voting system in [6]. It consists of two visual crypto layers showing the name of the voted candidate. A receipt is created by separating the layers and destroying one of them. Paul et al. [32] propose visual crypto for use in voter authentication for (non-cryptographic) remote voting systems. Scratch & Vote [1], Scratch, Click & Vote [24] and Pretty Good Democracy [38] make use of scratch-off coating to conceal encryption random factors and confirmation codes. Finally, Kelsey et al. [21] propose a voter-coercion strategy involving the use of scratch-off cards to direct voter action.

7 Future Work: Toward a Secure Multi-party Protocol

The systems described in this paper are both two-party protocols. Ultimately however it would be desirable to be able to distribute trust among arbitrarily many printers. With some modification the improved system presented in Section 4 could likely be extended to a secure multi-party protocol. With regard to creating the audit dataset this would be mostly a straightforward extension of the two-party approach with each of the $n > 2$ printers generating their own master permutations and issuing their own cut-and-choose proofs. Generating ballot tuples in a multi-party setting should also be a fairly straightforward extension of the two-party setting.

The primary challenge will be to develop an effective approach to distribute the ballot printing among more than two printers. This will undoubtedly require a fundamentally different approach from the two-party private printing scheme presented in [15] and is an interesting potential direction for future work.

Conclusion

Techniques for cryptographically verifiable elections offer unprecedented potential for making electronically-tabulated elections trustworthy. In this paper we presented two systems for cryptographically verifiable optical-scan voting that we believe offer properties that are desirable to both election officials and cryptographers. With this new approach election officials can continue to use a system with the familiar characteristics of optical-scan voting such as single layer ballots and paper audit trails which are both human-readable and conventionally auditable. Simultaneously the cryptographic audits can be conducted in a way that distributes trust such that no individual entity or piece of hardware has sufficient information to break voter privacy.

Acknowledgements

The authors wish to thank Jöern Mueller-Quade and the anonymous reviewers for their helpful feedback. Special thanks go to Jeremy Clark for many helpful discussions throughout the writing of this paper. This research is supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC)—the first author through a Postgraduate Scholarship and the third through a Discovery Grant. The second author was supported through a foreign exchange scholarship from the Karlsruhe House of Young Scientists (KHYS).

References

1. Ben Adida and Ronald L. Rivest. Scratch & vote: self-contained paper-based cryptographic voting. In *ACM WPES*, pages 29–40, 2006.
2. J. Benaloh. Administrative and public verifiability: Can we have both? In *EVT*, 2008.
3. Josh Benaloh. Ballot casting assurance via voter-initiated poll station auditing. In *EVT*, 2007.
4. Josh D. Benaloh (*né* Cohen) and Michael J. Fisher. A robust and verifiable cryptographically secure election scheme. In *SFCS*, 1985.
5. Richard T Carback, David Chaum, Jeremy Clark, John Conway, Aleksander Essex, Paul S. Hernson, Travis Mayberry, Stefan Popoveniuc, Ronald L. Rivest, Emily Shen, Alan T Sherman, and Poorvi L. Vora. Scantegrity II election at takoma park. In *USENIX Security Symposium*, 2010.
6. David Chaum. Secret-ballot receipts: True voter-verifiable elections. *IEEE Security and Privacy*, 2(1):38–47, 2004.
7. David Chaum, Richard Carback, Jeremy Clark, Aleks Essex, Stefan Popoveniuc, Ronald L. Rivest, Peter Y A Ryan, Emily Shen, and Alan T Sherman. Scantegrity II: end-to-end verifiability for optical scan election systems using invisible ink confirmation codes. In *EVT*, 2008.
8. David Chaum, Richard Carback, Jeremy Clark, Aleksander Essex, Stefan Popoveniuc, Ronald L. Rivest, Peter Y. A. Ryan, Emily Shen, Alan T. Sherman, and Poorvi L. Vora. Scantegrity ii: end-to-end verifiability by voters of optical scan elections through confirmation codes. *IEEE Transactions on Information Forensics and Security*, 4(4):611–627, 2009.

9. David Chaum, Aleksander Essex, Richard Carback, Jeremy Clark, Stefan Popoveniuc, Alan T. Sherman, and Poorvi Vora. Scantegrity: End-to-end voter verifiable optical-scan voting. *IEEE Security and Privacy*, 6(3):40–46, May/June 2008.
10. David Chaum, Peter Y A Ryan, and Steve Schneider. A practical voter-verifiable election scheme. In *ESORICS*, 2005.
11. Jeremy Clark and Urs Hengartner. On the use of financial data as a random beacon. In *EVT/WOTE*, 2010.
12. William Clarkson, T Weyrich, A Finkelstein, Nadia Heninger, J. Alex Halderman, and Edward W Felten. Fingerprinting blank paper using commodity scanners. In *IEEE Symposium on Security and Privacy*, 2009.
13. Ronald Cramer, Rosario Gennaro, and Berry Schoenmakers. A secure and optimally efficient multi-authority election scheme. In *EUROCRYPT*, 1997.
14. Aleks Essex, Jeremy Clark, Richard T. Carback, and Stefan Popoveniuc. Punchscan in practice: an e2e election case study. In *WOTE*, 2007.
15. Aleks Essex, Jeremy Clark, Urs Hengartner, and Carlisle Adams. How to print a secret. In *HotSec*, 2009.
16. Aleks Essex, Jeremy Clark, Urs Hengartner, and Carlisle Adams. Eperio: Mitigating technical complexity in cryptographic election verification. In *EVT/WOTE*, 2010.
17. Amos Fiat and Adi Shamir. How to prove yourself: practical solutions to identification and signature problems. In *CRYPTO*, pages 186–194, 1986.
18. Shafi Goldwasser, Yael Tauman Kalai, and Guy N Rothblum. One-time programs. In *CRYPTO*, 2008.
19. Richard T. Carback III, Stefan Popoveniuc, Alan T. Sherman, , and David Chaum. Punchscan with independent ballot sheets: Simplifying ballot printing and distribution with independently selected ballot halves. In *WOTE*, 2007.
20. Ayman Jarrous and Benny Pinkas. Secure hamming distance based computation and its applications. In *ACNS*, 2009.
21. John Kelsey, Andrew Regenscheid, Tal Moran, and David Chaum. Attacking paper-based E2E voting systems. In *Towards Trustworthy Elections*, volume 6000 of *LNCS*, pages 370–387. Springer, 2010.
22. Przemyslaw Kubiak. A modification of punchscan: Trust distribution. In *FEE*, 2006.
23. Ralf Küsters, Tomasz Truderung, and Andreas Vogt. Improving and simplifying a variant of prèt â voter. In *VOTE-ID*, 2009.
24. Miroslaw Kutylowski and Filip Zagorski. Scratch, click & vote: E2e voting over the internet. In *Towards Trustworthy Elections*. Spr, 2010.
25. D. Lundin, H. Treharne, P. Y. A. Ryan, S. Schneider, J. Heather, and Z. Xia. Tear and destroy: Chain voting and destruction problems shared by prèt â voter and punchscan and a solution using visual encryption. In *FEE*, 2006.
26. David Lundin and Peter Y. Ryan. Human readable paper verification of prèt â voter. In *ESORICS*, 2008.
27. Tal Moran and Moni Naor. Basing cryptographic protocols on tamper-evident seals. In *In Proceedings of the 32nd International Colloquium on Automata, Languages and Programming*, pages 285–297, 2005.
28. Tal Moran and Moni Naor. Split-ballot voting: Everlasting privacy with distributed trust. In *ACM CCS*, 2007.
29. Moni Naor and Adi Shamir. Visual cryptography. In *EUROCRYPT*, 94.
30. C. Andrew Neff. Practical high certainty intent verification for encrypted votes. Technical report, VoteHere Whitepaper, 2004.

31. Choonsik Park, Kazutomo Itoh, and Kaoru Kurosawa. Efficient anonymous channel and all/nothing election scheme. In *EUROCRYPT*, 1993.
32. Nathanael Paul, David Evans, Aviel D. Rubin, and Dan S. Wallach. Authentication for remote voting. In *HCISS*, 2003.
33. Torben Pryds Pedersen. A threshold cryptosystem without a trusted party. In *EUROCRYPT*, 1991.
34. Stefan Popoveniuc and Richard Carback. Clearvote: An end-to-end voting system that distributes privacy between printers. In *WPES*, 2010.
35. Stefan Popoveniuc and Ben Hosp. An introduction to punchscan. In *WOTE*, 2006.
36. Michael Rabin. Transaction protection by beacons. *Journal of Computer and System Sciences*, 27(2), 1983.
37. Peter Y A Ryan and Steve A Schneider. Prèt â voter with re-encryption mixes. In *ESORICS*, 2006.
38. Peter Y A Ryan and Vanessa Teague. Ballot permutations in prèt â voter. In *EVT/WOTE*, 2009.
39. Peter Y A Ryan and Vanessa Teague. Pretty good democracy. In *Workshop on Security Protocols*, 2009.
40. Daniel R. Sandler, Kyle Derr, and Dan S. Wallach. VoteBox: a tamper-evident, verifiable electronic voting system. In *USENIX Security Symposium*, 2008.
41. Alan T Sherman, Richard T Carback, David Chaum, Jeremy Clark, Aleksander Essex, Paul S. Hernson, Travis Mayberry, Stefan Popoveniuc, Ronald L. Rivest, Emily Shen, Bimal Sinha, and Poorvi L. Vora. Scantegrity mock election at takoma park. In *EVOTE*, 2010.
42. Zhe Xia, Steve A Schneider, and James Heather. Analysis, improvement and simplification of prèt â voter with paillier encryption. In *EVT*, 2008.