End-to-End Verifiability for Optical Scan Voting Systems

by

Emily Shen

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
[June 2008]
May 2008

Author .....
Department of Electrical Engineering and Computer Science
May 23, 2008

Certified by .....
Ronald L. Rivest
Viterbi Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Professor of Electrical Engineering
Chairman, Department Committee on Graduate Theses

End-to-End Verifiability for Optical Scan Voting Systems

by

Emily Shen

Submitted to the Department of Electrical Engineering and Computer Science
on May 23, 2008, in partial fulfillment of the
requirements for the degree of
Master of Science

Abstract

End-to-end verifiable voting systems allow voters to verify that their votes are cast as intended, collected as cast, and counted as collected. Essentially, end-to-end voting systems provide voters assurance that each step of the election worked correctly. At the same time, voting systems must protect voter privacy and prevent the possibility of improper voter influence and voter coercion. Several end-to-end voting systems have been proposed, varying in usability and practicality. In this thesis we describe and analyze Scantegrity II, a novel end-to-end verification mechanism for optical scan voting which uses confirmation codes printed on the ballot in invisible ink. The confirmation codes allow voters to create privacy-preserving receipts which voters can check against the bulletin board after the close of the election to ensure that their votes have been collected as cast. Anyone can check that votes have been counted as collected and that the tally is correct. We describe the Scantegrity II system and analyze the integrity and privacy properties it provides.

Thesis Supervisor: Ronald L. Rivest
Title: Viterbi Professor of Electrical Engineering and Computer Science

Acknowledgments

I would like to express gratitude to my advisor, Ron Rivest, for introducing me to voting and for his patient guidance and support throughout this research project.

I would also like to thank my long-distance collaborators, David Chaum, Richard Carback, Jeremy Clark, Aleks Essex, Stefan Popoveniuc, Peter Ryan, and Alan Sherman, for welcoming me to the Scantegrity II project and for their helpful discussions and insights.

Thanks also to my parents and sister and my friends for their encouragement and support.

Finally, I have been generously supported by the Bell Labs Graduate Research Fellowship Program.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The proper administration of elections is important to any modern democratic government. Yet, elections are notoriously difficult to get right. Voting is a complex system with strict requirements and constraints. History has shown numerous examples of electoral fraud [13]. The earliest evidence suggesting electoral fraud dates back to the first accounts of voting in ancient Greece, where pieces of pottery used as ballots were found with the same name written in the same handwriting [12]. Throughout more recent election history, there have been accounts of lost ballot boxes, ballot box stuffing, more than 100% voter participation, chain voting, vote selling, and other irregularities.

The highly controversial United States presidential election in Florida of 2000 exposed several basic problems with the way elections are run. In the 2000 election, an estimated 4 to 6 million presidential votes were lost (i.e., eligible voters wanting to vote could not vote or their votes were not counted for some reason) [22]. This number includes 1.5 to 2 million votes lost because of faulty voting equipment and confusing ballots such as the "butterfly ballot", 1.5 to 3 million lost because of voter registration problems, up to 1 million lost because of problems with polling place operations such as lines, hours, and locations, and an unknown number of votes lost because of problems with absentee ballots. Problems such as these, especially faulty voting equipment and confusing ballots, prevent voters from being confident in the electoral process and in the integrity of the election outcome. With the 2008 U.S.

presidential election, there is renewed interest in addressing these issues.

While there are many important issues related to elections, including voter registration, polling place operations, absentee ballots, and voting equipment, in this thesis we will be concerned with the design of voting systems which allow voters to be confident in the outcome of elections. Although the term "voting system" has been used to refer to the method of tallying votes and determining the winner(s) of an election (e.g., first-past-the-post) [1], in this thesis we will use it to refer to the technology and process used for voting and for verifying the results.

1.1 Voting Technologies

There are five major voting technologies in use in the U.S. today (see [22] for a more detailed description). Three of these technologies are based on paper ballots: hand-counted paper ballots, punch cards, and optically scanned paper ballots. Hand-counted paper ballots are the oldest voting technology in use in the U.S. and have the disadvantage that counting is too slow for large-scale elections.

Punch cards, introduced in the 1960s, and optical scan, made prevalent in the 1990s, automate the counting. Punch cards require the voter to indicate her choices by making holes next to her chosen candidates in the ballot card using a stylus. One problem with punch cards is that hole punches can be imprecise and leave bits of "chad" which make it difficult to determine the voter's intent. Also, when used with "butterfly" layouts that use both the left and right sides of the page, the system can be confusing to voters.

Optical scan ballots require the voter either to complete an arrow by connecting two points, or to fill in a bubble, as in standardized tests. There are two types of counting for optical scan ballots: central-count and precinct-based. In a central-count system, ballots are cast at the precinct and then processed and scanned at a central location. In a precinct-based system, voters feed their ballots into a scanner at the precinct and get immediate feedback on any marking errors. The scanner will return the ballot to the voter if there are any errors such as stray marks or overvotes (i.e.,

votes for more than one candidate for the same race), and the voter will get a chance to correct these errors. For this reason, precinct-based scanning reduces the number of lost votes [15].

Two other voting technologies are in use: lever machines and Direct Recording Electronic (DRE) machines. The lever machine, introduced in the late nineteenth century, consists of a set of levers associated with each candidate on the ballot which the voters pull to indicate their choices. DRE voting machines present the ballot and feedback information on a touchscreen interface and may also include audio feedback. DRE machines have several advantages. They can prevent a voter from overvoting (as with precinct-based optical scan) and warn a voter if she is undervoting (i.e., not selecting any candidate). Ballots can more easily be provided in multiple languages. Visually impaired voters can vote without assistance using the audio instructions and feedback.

However, DRE machines have posed significant security concerns because they do not provide voters assurance that they are recording votes properly. Firstly, code for DRE machines must be tested and certified, but certification does not guarantee that the code is correct. Secondly, even if the certified code is correct, voters cannot verify that the certified code is what is running on the machine on the day of the election.

To provide some form of voter verification, a voter-verified paper audit trail (VVPAT) has been proposed [17]. In a VVPAT system, after the voter enters her choices, the machine will print and display a receipt of the ballot behind glass. The voter will have the option of confirming or canceling her vote. If the voter confirms her vote, the ballot receipt is deposited into a sealed ballot box. This method allows the paper ballots to be used in the case of a recount.

1.2 Voting System Requirements

To understand what makes voting a hard problem, we consider the requirements for voting systems. We will focus on the requirements of integrity and privacy. We refer the reader to [2] for a more thorough review of voting system requirements and

end-to-end voting systems.

1.2.1 Integrity and Verifiability

Each voter's vote should be cast, collected, and counted as intended. We separate this into three properties: cast as intended, collected as cast, and counted as collected. In order to provide voters confidence in the integrity of elections, the integrity of the election should be verifiable by voters.

1.2.2 Privacy and Incoercibility

At the same time, we require that the voting system maintain voter privacy. That is, no one else besides the voter herself should know how an individual voter voted, even if the voter wishes to reveal this information. The voting system must not allow a voter to prove how she voted because this would allow vote selling and voter coercion. Therefore, the voter must not be able to prove to anyone how she voted.

1.2.3 Verifiability vs. Privacy

These requirements seem apparently contradictory. Consider a hypothetical voter Valerie. Valerie should be able to receive assurance of the integrity of the election, in particular that her vote was counted properly. We want Valerie to be able to take away some information such as a receipt of her vote, but this receipt must not reveal how Valerie voted. The voting and verification processes must not reveal how Valerie voted, even if Valerie is willing to cooperate with a coercer.

Since the introduction of the secret ballot, voting systems have generally emphasized privacy over verifiability. Private polling booths and sealed ballot boxes ensure vote secrecy. Various steps of the election process are verifiable to some degree. Voters can visually verify visually that their votes are cast as they intend, observers are allowed to be present at the polling place and at the counting facility, and post-election audits can verify that the paper ballots do not differ significantly from the electronic tally.

However, we can improve on this degree of verifiability. With any of the voting technologies in use today, it is necessary to trust the election officials and the physical chain of custody of the ballots. Voters do not receive any assurance that their votes were not lost or modified and that all cast ballots are counted. Voters have to trust that the equipment has been programmed and tested properly. We would like the ability to verify the integrity of the election to be available to all voters and interested parties.

1.3 End-to-End Verifiable Voting Systems

To provide voters assurance that every step of the election worked correctly, new voting systems called end-to-end (E2E), or cryptographic, voting systems, have been proposed. End-to-end verifiability allows voters to verify that their votes are cast, collected, and counted as intended. Specifically, individual voters can verify that their votes are cast as intended and collected as cast (voter verifiability), and any party can verify that the votes are counted as collected (universal verifiability). Thus, voters gain assurance that every step of the voting process is executed properly. At the same time, end-to-end voting systems are required to maintain voter privacy; that is, the voting and verification mechanisms do not reveal how an individual voter voted. End-to-end voting systems preserve the secrecy of traditional secret ballot elections while additionally allowing verifiability of the election integrity.

Typically, cryptographic voting schemes use a public bulletin board where election officials publish information that voters and others use for the verification process. At the time of voting, voters can optionally receive a privacy-preserving receipt of their vote. After the votes are counted, the results and verification information are posted on the bulletin board. Voters can use their receipts to verify that their votes have been collected as cast, and any party can verify that the tally is correct with respect to the collected votes. If any voter finds a discrepancy, she has some time to file a dispute before the results are certified.

It is important to consider the recovery measures to adopt if the verification

mechanism reveals a problem. We require that voters must be able to file disputes in such a way that does not compromise their vote secrecy. In addition, we would like the voting and verification mechanisms to be easy to use and impose as little burden as possible on voters.

1.3.1 Verifiability vs. Incoercibility

With classical voting systems, secrecy implies incoercibility. That is, if the voting system does not reveal how any individual voter voted, then that voter cannot reliably be coerced into voting a particular way.

However, with cryptographic voting systems, a new type of coercion threat is possible. These voting systems often use some kind of randomization (for example, a randomized candidate ordering), and in some cases it may be possible for a coercer to coerce a voter into voting for a random candidate. Thus, we require a stronger property than ballot secrecy. We require that a coercer cannot have any enforceable strategy to influence a voter's vote in any way. In particular, a coercer cannot force a voter to vote for a random candidate or to submit an undervote for a particular race.

1.4 Contribution

Several end-to-end voting schemes have been proposed, most of which use some form of cryptography to achieve both verifiability and privacy. Previously proposed schemes include SureVote [6], Prêt-à-Voter [9], Scratch&Vote [3], Punchscan [11, 21], Scantegrity [20, 8], and ThreeBallot [23]. These schemes vary in usability and practicality, and some are open to some form of randomizing coercion attack.

In this thesis we describe and analyze Scantegrity II, a joint project with David Chaum, Richard Carback, Jeremy Clark, Aleksander Essex, Stefan Popoveniuc, Ronald L. Rivest, Peter Y. A. Ryan, and Alan T. Sherman, to appear at the USENIX/ ACCURATE Electronic Voting Workshop in July 2008 [7].

Scantegrity II is a novel end-to-end verifiability mechanism for optical scan voting systems that uses confirmation codes printed in invisible ink on ballots. To vote for

a candidate, the voter uses a special decoder pen to mark the bubble next to the candidate's name. Marking the bubble with the decoder pen simultaneously reveals a previously hidden confirmation code printed inside the bubble and leaves a dark mark in the bubble that is recognizable by an optical scanner. The voter can copy down the revealed confirmation codes corresponding to her candidate choices onto her receipt. The voter then detaches the receipt from the ballot and feeds the ballot into the scanner. After election day, election officials post the revealed confirmation codes on the web bulletin board and voters can check their receipts against the bulletin board to check that their votes were collected as cast. Additionally, any interested party can check that the votes were counted as collected. If any voters find discrepancies between the confirmation code they copied down and those posted on the bulletin board, they can file disputes, which are handled through a dispute resolution protocol. In the original Scantegrity system, the dispute resolution protocol was rather complicated in order to preserve voter privacy. In Scantegrity II, the invisible ink confirmation codes make the dispute resolution protocol much simpler.

Scantegrity II provides the following advantages over current voting systems:

- Compatibility with optical scan equipment
- A familiar ballot marking procedure
- Immunity to randomization and other coercion attacks
- A privacy-preserving mechanism for handling disputes

In Chapter 2, we survey related paper-based cryptographic voting schemes. In Chapter 3, we describe the Scantegrity II system. In Chapter 4, we analyze the security properties of the system. We conclude in Chapter 5.

Chapter 2

Related Work

In this chapter we cover preliminaries and survey some paper-based end-to-end voting schemes related to Scantegrity II. We will consider the following schemes: Prêt-à-Voter [9], Scratch&Vote [3], ThreeBallot [23], Punchscan [11, 21], and Scantegrity [8, 20].

2.1 Preliminaries

In this section we cover some cryptographic tools that are used in Scantegrity II as well as in several other end-to-end verifiable voting systems.

2.1.1 Commitment Schemes

A commitment scheme [19] allows a user to commit to a value so that the value is hidden and the user cannot change the value but can reveal the committed value later. A commitment consists of a commit phase and then a reveal phase. The properties we require of a commitment scheme are that it be binding (only the committed value can be revealed later) and hiding (the committed value cannot be determined from the commitment). Commitment schemes can be either perfectly binding and computationally hiding or perfectly hiding and computationally binding. In practice, in Scantegrity II, we implement a commitment scheme as a randomized hash function.

2.1.2 Secret Sharing

Secret sharing schemes allow a secret value to be distributed among individuals such that each individual receives a share of the secret. The secret can be reconstructed only when the shares are combined. In general, a (t, n) secret sharing scheme allows any subset of t of the n participants to reconstruct the secret, but any subset of size less than t cannot gain any information about the secret. The first secret sharing schemes were proposed independently by Shamir [24] and Blakley [4].

2.2 Paper-Based End-to-End Voting Systems

In this section we briefly discuss some paper-based end-to-end voting systems.

2.2.1 Prêt-à-Voter

The Prêt-à-Voter [9] ballot is separated into two parts. On the left half of the ballot are the candidate names, in a randomized order. On the right half are blanks for the voter to mark her choices. The voter marks the blank corresponding to her choice and then separates the two halves of the ballot and casts the right half. A copy of the right half becomes the voter's receipt. On the right half of the ballot is printed an "onion", a value which allows election officials to recreate the candidate ordering in order to decrypt the vote.

2.2.2 Scratch&Vote

The Scratch&Vote [3] ballot is based on the Prêt-à-Voter ballot. Like the Prêt-à-Voter ballot, on the left half of the ballot are the candidate names, in a randomized order. On the right half are blanks for the voter to mark her choices. Also on the right half are a barcode and a scratch surface. The scratch surface covers information necessary to audit the correct formulation of the ballot. The barcode encodes ciphertexts used for homomorphic aggregation of votes. To vote for a candidate, the voter marks the blank corresponding to her choice and then separates the two halves of the ballot.

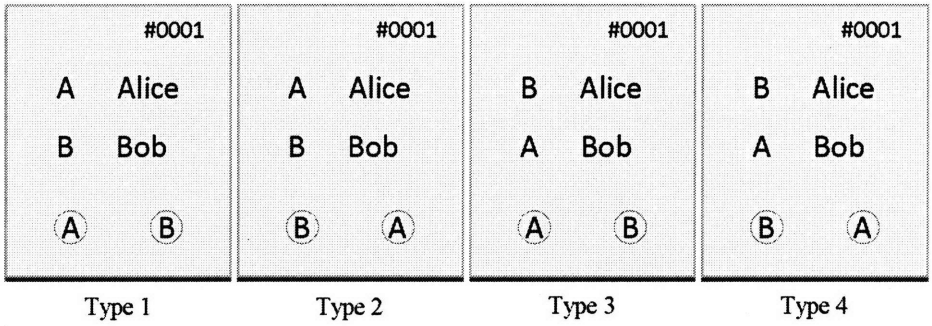

Figure 2-1: The four possible types of Punchscan ballots (unmarked) for a single-race election between two candidates, Alice and Bob.

The voter then presents the ballot to the election official, who checks that the scratch surface is intact and then destroys the scratch surface. The voter then feeds the remainder of the right side of her ballot into the scanner and takes this home as her receipt.

2.2.3 Punchscan

In Punchscan [11, 21], the ballot is composed of two superimposed sheets, a top sheet and a bottom sheet. The top sheet contains a list of candidate names, with a code next to each candidate name. For example, if a race consists of four candidates, the codes might be A, B, C, and D. The codes appear next to the candidate names in a randomized order. The top sheet also contains circular holes (the same number as there are candidates) letting the bottom sheet show through. On the bottom sheet, positioned under the holes from the top sheet are the same codes as on the top sheet but in an independent randomized order. Figure 2-1 shows the four possible types of ballots for a single-race election between two candidates, Alice and Bob.

To vote for a candidate, the voter finds the code on the top sheet that appears next to the candidate name. Then the voter finds the same code on the bottom sheet through the holes and marks the corresponding hole using a "bingo dauber" which leaves a mark on both the top sheet and the bottom sheet. The voter arbitrarily selects either the top sheet or the bottom sheet as her receipt. Figure 2-2 shows a sample Punchscan ballot for a single-race election between Alice and Bob, with a vote

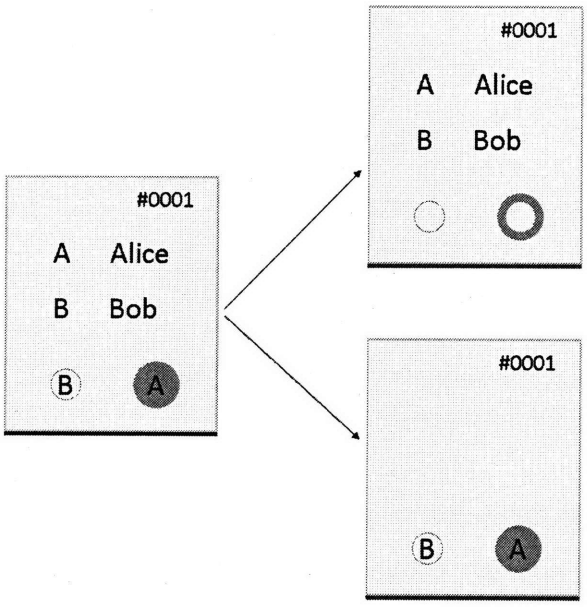

Figure 2-2: A Punchscan ballot with a vote marked for Alice. The left shows the ballot with the top and the bottom sheet superimposed. The right shows the top and the bottom sheet separated.

marked for Alice. In the earlier version of Punchscan, the voter was free to choose her receipt sheet after viewing and voting on the ballot. However, this aspect made Punchscan vulnerable to a coercion attack, which we will describe later, and the latest version of Punchscan requires the voter to choose which sheet to keep as her receipt before receiving the ballot.

2.2.4 Scantegrity

Scantegrity [8, 20] is an improvement on Punchscan which combines the randomization of the two sheets into one sheet. The ballot looks like a conventional optical scan ballot except that next to each candidate name appears a code. The codes appear in a randomized order. Figure 2-3 shows a sample Scantegrity ballot for a single-race election between Alice and Bob. To vote for a candidate, the voter marks the bubble next to her chosen candidate and notes the letter that appears next to the bubble. After the end of election day, the letters next to the marked bubbles are posted for each ballot on the bulletin board. If a voter finds that a posted letter is incorrect, she can file a dispute. However, the dispute resolution protocol is rather complicated

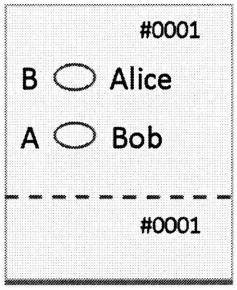

Figure 2-3: A Scantegrity ballot.

in order to preserve voter privacy.

2.2.5 ThreeBallot

The ThreeBallot proposal [23] is unique among end-to-end voting systems in that it does not use any cryptography and is purely paper-based. Each voter receives a "multi-ballot" which consists of three conventional ballots, each with a unique, random ballot ID. To vote for a candidate, the voter marks the bubble next to the candidate's name on exactly two of the three ballots. For every other candidate, the voter marks the bubble next to the candidate's name on exactly one ballot. The voter feeds the multi-ballot into a scanner-checker, which checks that the multi-ballot has been filled out properly. If so, the voter is allowed to arbitrarily choose one of the three ballots as her receipt. The scanner returns a copy of the chosen ballot to the voter and the voter casts the three ballots.

Afterwards, election officials post electronic ballot images for all cast ballots. Each voter can check that there is an image which matches her receipt. The tally can be computed by anyone by counting the number of marks for each candidate (the number of votes for each candidate is increased by the number of voters).

2.3 Coercion Attacks

All of the schemes described above protect voter privacy in the sense that a voter's receipt does which candidate she voted for. However, as discussed in section 1.3.1, secrecy of a voter's candidate choices does not imply incoercibility; that is, the voting

system may be vulnerable to vote coercion attacks, including coerced randomization. Even if the receipts do not reveal information about how the voter voted, without the voter's cooperation, it is possible to coerce a voter to vote a certain way with the voter's cooperation. In this section, we will describe a few coercion attacks against end-to-end voting systems in order to illustrate the difficulty of protecting against coercion attacks even when receipts do not reveal voter's candidate choices. For a more detailed description of coercion attacks against end-to-end voting systems, see [16].

2.3.1 Coerced Randomization

A well-known issue with many end-to-end voting systems is that a coercer can randomize a voter's vote by instructing her to bring back a receipt that shows, for example, the first candidate choice selected. Specifically, for Pîret-à-Voter and Scratch&Vote the voter can be instructed to bring back a receipt with a mark in the first blank, which corresponds to a random candidate. For Punchscan, the voter is instructed to bring back a receipt with a mark for the first hole. For Scantegrity, the voter is instructed to vote for the candidate with the letter "A" next to the name, and the coercer can check that this letter appears on the receipt posted on the bulletin board with the voter's ballot ID.

2.3.2 Incentive Attacks

In some cases it is possible that even if a coercer does not learn how a voter voted, the coercer can provide an incentive for voters vote for a particular candidate. We will describe this type of attack against the earlier version of Punchscan, in which voters were allowed to choose their receipt sheet after viewing and voting the ballot. The Punchscan protocol has since been modified to require voters to choose their receipt sheet before viewing the ballot, but we will describe attacks against the earlier version for the purposes of illustrating how these incentive attacks might work.

As a concrete example, consider a single-race election with two candidates, Alice

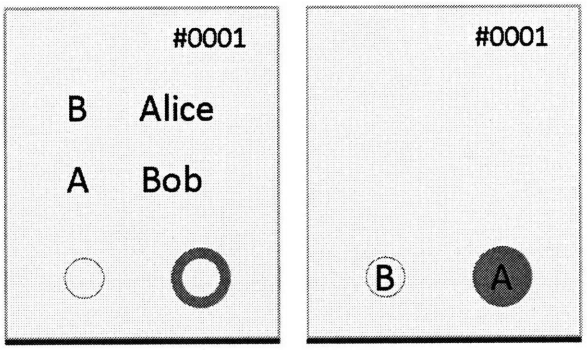

Figure 2-4: A coercer can coerce a voter to vote for Alice by requiring that the voter bring back a receipt that is NOT one of these two.

and Bob. Suppose a coercer wishes to influence the election in favor of Alice. A simple coercion strategy by Moran and Naor described in [18] forces voters to vote for Alice 1/4 of the time and allows voters to vote as they choose 3/4 of the time. A similar coercion strategy presented by Bohli et al. in [5] pays voters so that voters will be paid for voting for Alice 1/2 of the time, voters will be paid for voting for Bob 1/4 of the time, and 1/4 of the time voters will not be able to bring back a receipt that allows them to be paid and therefore can vote as they choose. We will describe the coercion attack of Moran and Naor below.

To coerce voters to vote for Alice, the coercer requires that the voter bring back a receipt that is not one of the two shown in Figure 2-4. Consider the four ballot types shown in Figure 2-1. If the voter receives a Type 1, 2, or 3 ballot, she can vote as she chooses, submitting either the top or the bottom sheet for a Type 1 ballot, the top sheet for a Type 2 ballot, and the bottom sheet for a Type 3 ballot. However, if the voter receives a Type 4 ballot, the voter must vote for Alice in order to bring back a receipt that is not one of the two shown in Figure 2-4. Thus, with 1/4 probability a voter will be forced to vote for Alice.

Kelsey et al. [16] extend this strategy to the case of multiple candidates to influence the vote against a particular candidate by creating different levels of payment for different receipts, effectively randomizing the vote away from a particular candidate. This strategy works under the assumption that voters will act to maximize their payout.

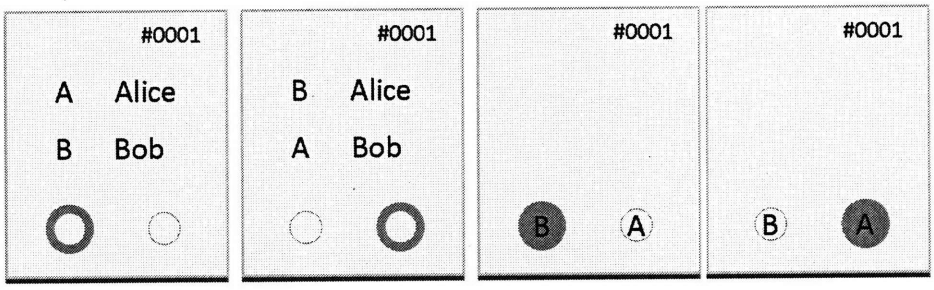

Figure 2-5: A coercer can coerce a voter to vote for Alice by requiring that the voter bring back a receipt that is of one of these four types.

In fact, we can come up with a coercion attack which works with 1/2 probability. This attack works as follows. To coerce voters to vote for Alice, the coercer requires that the voter bring back a receipt that is one of the four shown in Figure 2-5. If the voter receives a Type 2 or Type 4 ballot, she can vote as she chooses, submitting the bottom sheet. However, if the voter receives a Type 1 or Type 3 ballot, the voter must bring back the top sheet and, therefore, must vote for Alice. Thus, with 1/2 probability a voter will be forced to vote for Alice.

The coercion and incentive attacks described above illustrate the point that vote privacy is not equivalent to vote incoercibility. That is, even if a voting system does not allow a voter to prove which candidate she voted for, it may still be vulnerable to more subtle types of coercion attacks.

Scantegrity II thwarts coercion and incentive attacks because a voter receives no information (such as confirmation codes) specific to her ballot, besides her ballot ID, before indicating her candidate choices. Therefore, a voter cannot be coerced into bringing back a receipt with a particular property.

Chapter 3

Scantegrity II

This chapter and the next describe work to appear at the USENIX/ACCURATE Electronic Voting Technology Workshop in July 2008 [7]. This is joint work with David Chaum, Richard Carback, Jeremy Clark, Aleks Essex, Stefan Popoveniuc, Ronald L. Rivest, Peter Y. A. Ryan, and Alan T. Sherman.

3.1 Overview

Scantegrity II (Invisible Ink) is an end-to-end verifiability mechanism for optical scan voting that enables each voter to verify that her vote is collected as cast and enables any interested party to verify that all votes are counted as collected. The Scantegrity II ballot is similar to a conventional optical scan ballot, with extra information printed in *invisible ink* to allow voters to create privacy-preserving receipts of their votes. The end-to-end verifiability capability is opt-in; that is, voters may choose not to participate and their voting experience will be largely the same as with a conventional optical scan system. The integrity of the system depends on only a minimally sufficient number of voters participating in voter verification.

The contributions of Scantegrity II can be summarized as follows:

- Compatibility with optical scan technology: Scantegrity II works with existing optical scan polling place equipment. It interfaces cleanly with the underlying

system, only requiring a modified ballot and access to the results from the scanners.

- Familiar ballot marking procedure: The voter marks the bubble next to the candidate she wishes to vote for, just as on a conventional optical scan ballot, but using a special *decoder pen.*

- Immunity to randomization attacks: When a voter looks at an unmarked Scantegrity II ballot, no information other than the candidate choices is visible. Therefore, a coercer cannot force a voter to mark her ballot in a way that creates a specified type of receipt, in contrast to many other end-to-end voting systems.

- Improved mechanism for handling disputes: Invisible ink and cryptographic commitments are used to commit to information on the ballots so that election officials can distinguish between serious claims of discrepancies and spurious ones. Statistical triggers can be set to allow fraud to be detected while minimizing the false positive rate.

3.1.1 Voter Experience

We now give a high-level description of the voter experience.

- Check-in: The voter checks in and asks for either one ballot to vote on or two ballots – one to vote on and one to audit. If she chooses to receive two ballots, she chooses one arbitrarily to audit according to a procedure to be described in section 3.6. The voter is given a special decoder pen and a regular pen.

- Indicating her choices: To indicate a vote for a candidate, the voter marks the bubble next to that candidate's name using the decoder pen, revealing a previously hidden confirmation code in that bubble.

- Creating a receipt: Using the regular pen, the voter may choose to note the revealed confirmation codes in the receipt portion of the ballot, if she wishes to participate in post-election voter verification.

- Detaching the receipt: The voter detaches the receipt from the ballot along a perforation.

- Casting the ballot: The voter feeds her ballot into the optical scanner.

- Post-election voter verification: Within an announced time frame after results are posted on the web bulletin board, the voter may check that the confirmation codes revealed on her ballot are posted correctly. The voter may also check that her audited ballot was constructed properly. Any interested party may check the correctness of the tally with respect to the posted confirmation codes. If there are any discrepancies, voters may file disputes, which will be handled by election officials in a manner to be described in sections 3.9 and 3.10.

3.2 The Ballot

The Scantegrity II ballot is an extension of a conventional optical scan ballot. The ballot consists of a *voting portion* and a *receipt portion*, each printed with the ballot's ID number.

Each ballot has a distinct *ballot ID*. The ballot ID uniquely specifies the election and the ballot within the election. An example ballot ID might be "MA-2008-11-04-123456789". For convenience, we will use 4-digit ballot ID's in the examples that follow.

The voting portion contains the list of candidate names for each race. Next to each candidate name is an optical mark recognition field, referred to as a *bubble*. Each bubble contains a hidden random sequence of alphanumeric characters, referred to as a *confirmation code*, printed in invisible ink. Figure 3-1 (left) shows a sample ballot image file for the printer for ballot #0001 in a single-race election with two candidates, Alice and Bob. Before the ballot is marked, all of the confirmation codes are hidden, as shown in figure 3-1 (middle). To indicate a vote for a candidate, the voter marks the bubble for that candidate with a special *decoder pen*. The ink in the decoder pen reacts with the invisible ink printed inside the bubble to simultaneously

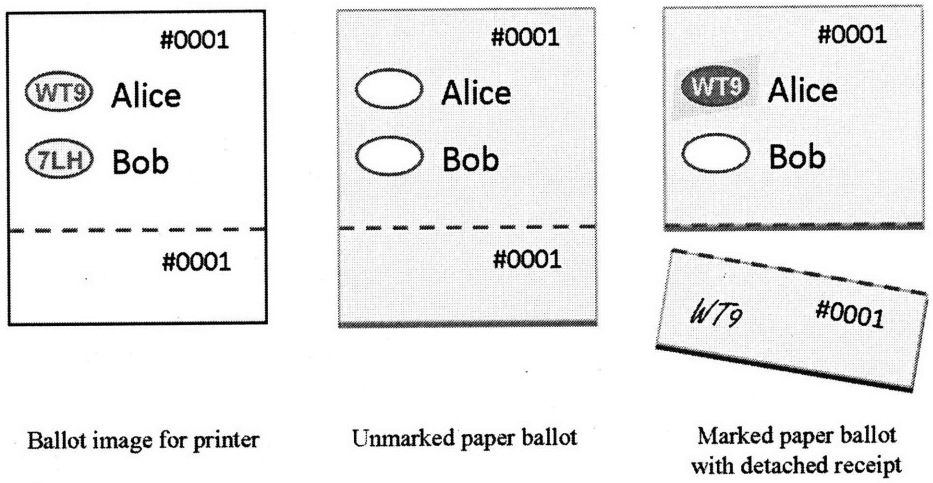

Figure 3-1: **Example Ballot.** Left: Ballot image for printers. Middle: Printed paper ballot. The confirmation codes, printed in invisible ink, are initially hidden. Right: Marked paper ballot and detached receipt. Marking a bubble with the decoder pen reveals the confirmation code and leaves a dark mark inside the bubble. The voter may note the revealed confirmation code on her receipt.

reveal the confirmation code and leave a dark mark in the bubble that is recognizable by an optical scanner. Figure 3-1 (right) shows the confirmation code revealed by a vote for Alice on ballot #0001.

The receipt portion of the ballot contains the list of race titles (if the ballot contains multiple races), with space next to each title for the voter to optionally note the confirmation codes revealed by her choices. The receipt portion is located across the bottom of the ballot and is detachable by the voter along a perforation. Figure 3-1 (right) shows a voter's detached receipt for ballot #0001 with the revealed confirmation code noted.

3.3 Confirmation Codes

The confirmation codes have the following properties:

- The confirmation codes are unique within each race on each ballot.
- The confirmation codes are uniform-pseudorandomly selected from the set of possible codes.

- The assignment of confirmation codes to candidates on each ballot is pseudo-random.

- The confirmation code corresponding to a particular candidate on a particular ballot is hidden (by invisible ink) from the voter unless and until she marks the bubble for that candidate.

- The set of alphanumeric characters that may be used in confirmation codes is restricted to eliminate those characters which are most commonly confused (e.g., 0 (zero) and O (oh), 1 (one) and I (eye), etc.).

The invisible ink acts as a kind of commitment to the confirmation codes on the ballots in that it is both hiding and binding. The same confirmation codes will also be committed to cryptographically on the web bulletin board by election officials (see section 3.4). We will refer to the confirmation codes printed on the ballots as *paper codes* and those posted on the web bulletin board as *web codes.* Auditing of the ballots (see section 3.6) ensures that the paper codes for each ballot match the web codes for that ballot. Drawing on the analogy between the invisible ink "commitments" and the cryptographic commitments, we will use the terms *hidden* and *revealed* to refer to the states of both the paper codes and the web codes. Paper codes revealed with the decoder pen will be called *paper-revealed codes* and those that remain hidden will be called *paper-hidden codes.* Web codes whose cryptographic commitments are opened will be called *web-revealed codes* and those that remain unopened will be called *web-hidden codes.*

3.4 Election Preparation

For simplicity, we consider a single-race election. Let N be the number of candidates and let B be the number of ballots printed. The N candidates are assigned an arbitrary ordering.

Prior to election day, the election officials secret share a seed to a pseudorandom number generator (PRNG). Election officials then use a trusted workstation, as

described in [11], to jointly generate $B \cdot N$ pseudorandom confirmation codes and construct the following tables:

$\mathbf{P} := (p_i)_{B \cdot N}$: Table **P** contains the confirmation codes in the order generated by the PRNG. Table **P** specifies the assignment of confirmation codes to candidates on each ballot. The codes are grouped by ballot ID. The code for candidate k on ballot j is $p_{(N(j-1)+k)}$, i.e., codes $p_1, ..., p_N$ correspond to candidates $1, ..., N$, respectively, on ballot #1, $p_{N+1}, ..., p_{2N}$ correspond to candidates $1, ..., N$, respectively, on ballot #2, and so on. Table **P** is used to generate the ballot image files for the printers and to generate table **Q**. Table **P** is never published.

$\mathbf{Q} := (q_i)_{B \cdot N}$: Table **Q** contains the confirmation codes from table **P**, pseudorandomly permuted within each ballot ID, i.e., for each ballot j, codes $p_{(N(j-1)+1)}, \cdots, p_{(N(j-1)+k)}$ are pseudorandomly permuted to obtain codes $q_{(N(j-1)+1)}, \cdots, q_{(N(j-1)+k)}$. Election officials commit to each code in table **Q** and publish these commitments on the web bulletin board.

$\mathbf{R} := (r_{i,j})_{B \cdot N \times 3}$: In table **R**, each row corresponds to a distinct confirmation code from table **Q** and contains a *flag* and two *pointers* for that code. The order of the underlying confirmation codes in **R** is a pseudorandom permutation of all of the codes in **Q**. The flag for a code will be raised during the post-election posting phase (see section 3.7) if the code is paper-revealed on a voted ballot. Each row contains two pseudorandom pointers – a ***Q**-pointer* specifying the index of a code in table **Q** and a ***S**-pointer* specifying the index of a flag in table **S** (described below). Election officials commit to each pointer in table **R** and publish these commitments on the web bulletin board.

$\mathbf{S} := (s_i)_{B \cdot N}$: In table **S**, each row corresponds to a distinct confirmation code from table **Q** and contains a flag which will be raised if the code is paper-revealed on a voted ballot. The flags are grouped by candidate, and the order of the underlying confirmation codes within each candidate block is a pseudorandom permutation of the codes for that candidate. Table **S** (initially empty) is published on the web bulletin board.

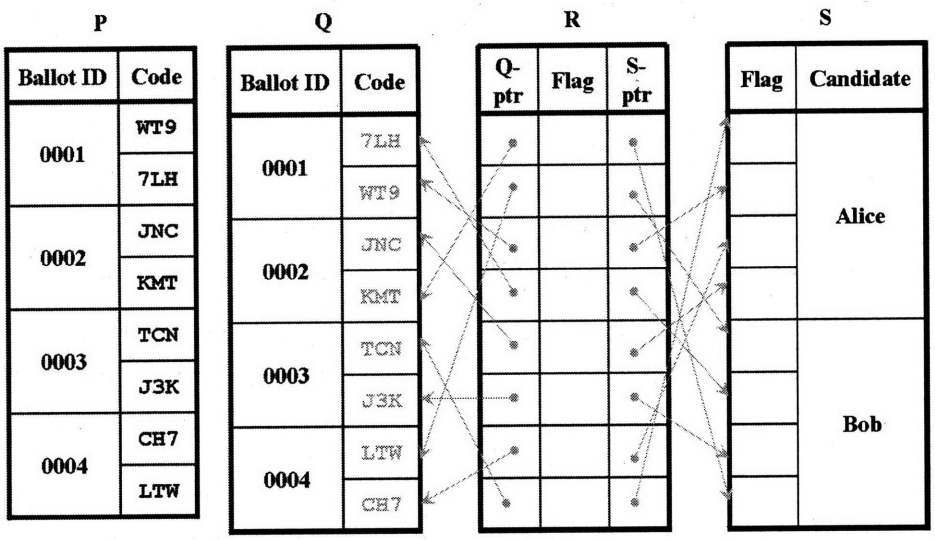

Figure 3-2: Tables **P**,**Q**, **R**, and **S** generated by election officials before election day. Table **P** is used for ballot printing and is never published. Tables **Q**, **R**, and **S** are published on the web bulletin board. Entries shown in gray are visible only to election officials; the web bulletin board displays commitments to these entries. For example, a vote for Alice on ballot #0001 would reveal the confirmation code `WT9`. The corresponding row of table **R** is row 3, which points to ballot #0002 of table **Q** and to row 2 of table **S**, which corresponds to a vote for Alice.

To summarize, table **Q** contains the confirmation codes, organized by ballot ID but pseudorandomly permuted within each ballot ID block to hide the correspondence to the candidates. Table **S** contains flags for the confirmation codes, organized by candidate but pseudorandomly permuted within each candidate block to hide the correspondence to ballot ID's. Table **R** provides a pseudorandom permutation of all the confirmation codes and contains pointers to the corresponding entries in **Q** and **S**. Table **R** will be used to audit the tables via randomized partial checking [14].

As a working example, consider a single-race election with $N = 2$ candidates, Alice and Bob, and $B = 4$ ballots. Let Alice be candidate #1 and Bob be candidate #2. Figure 3-2 shows table **P** (unpublished) and tables **Q**, **R**, and **S** as published on the web bulletin board before election day. Entries shown in gray are visible only to election officials; the web bulletin board displays commitments to these entries.

3.5 Voting

After checking in, the voter chooses to receive either one ballot to vote on or two ballots – one to audit and one to vote on. If she chooses two ballots, she chooses one of the two arbitrarily to audit (as described in section 3.6) and votes on the other one. The voter is also given a decoder pen and a regular pen.

To vote for a candidate, the voter marks the bubble next to that candidate's name using the decoder pen, simultaneously revealing the paper code and leaving a dark mark inside the bubble that is recognizable by an optical scanner.

If the voter wishes to participate in post-election verification that her vote is collected as cast, she may note the paper-revealed codes in the receipt portion of the ballot, using the regular pen. The voter detaches the receipt portion of the ballot.

The voter then feeds her ballot into the optical scanner, which checks that the ballot has been properly marked (i.e., there are no stray marks, overvotes, etc.). If there is an error, the scanner rejects the ballot and the voter may obtain a new ballot to vote on. Otherwise, the scanner accepts the ballot and records the voter's candidate choices along with the ballot ID.

3.6 Ballot Auditing

Ballots are audited to ensure that they are printed correctly with respect to the tables committed to on the web bulletin board. Ballot auditing should be done at all points during the election process. Candidates can pre-audit a random sample of ballots prior to election day. On election day, voters may obtain two ballots and audit one and vote on the other. At the end of the day, any remaining blank ballots should be audited.

When a ballot is audited, it is stamped "Official Audit Ballot". The voter auditing the ballot reveals all of the codes on the ballot using a decoder pen. The voter checks that the codes are unique within each race. After election day, election officials web-reveal all entries associated with audited ballots, i.e., the confirmation codes in $\mathbf{Q}$ and

the **Q**- and **S**- pointers for the rows in **R** corresponding to those codes. The voter then checks that the web-revealed codes are the same as the paper-revealed codes, and that the web-revealed pointers map each code in **Q** to the appropriate candidate block in **S**.

3.7 Posting and Tallying

After the close of the election, election officials jointly regenerate table **P** and use a copy of the electronic ballot images from the optical scanners to translate the votes into the paper-revealed codes. Election officials web-reveal those codes in table **Q** and flag the entries corresponding to those codes in tables **R** and **S**. The tally for each candidate is computed by counting the number of flags in the candidate's block in **S**. Election officials also publish the official tallies reported by the optical scanners and a list of the voters who voted.

To enable the auditing of ballots, election officials web-reveal all of the codes for those ballots which were recorded as audited. To enable the auditing of the tables via randomized partial checking (RPC) [14], for each entry in **R**, either the **Q**-pointer or the **S**-pointer is web-revealed (unless both were already revealed, for an audited ballot), depending on a pseudorandom, publicly verifiable coin flip.

In our working example, suppose that ballots #0001 and #0003 were voted for Alice, ballot #0002 was voted for Bob, and ballot #0004 was spoiled for audit. Figure 3-3 shows tables **Q**, **R**, and **S** as published on the web bulletin board after election day.

3.8 Checking Receipts

After the end of the election, a voter who has made a receipt of her paper-revealed codes can go to the web bulletin board and look up her ballot ID. She checks that in the block of **Q** corresponding to her ballot ID, all of her paper-revealed codes, and only these, are web-revealed. Anyone can check that the commitments are correct.

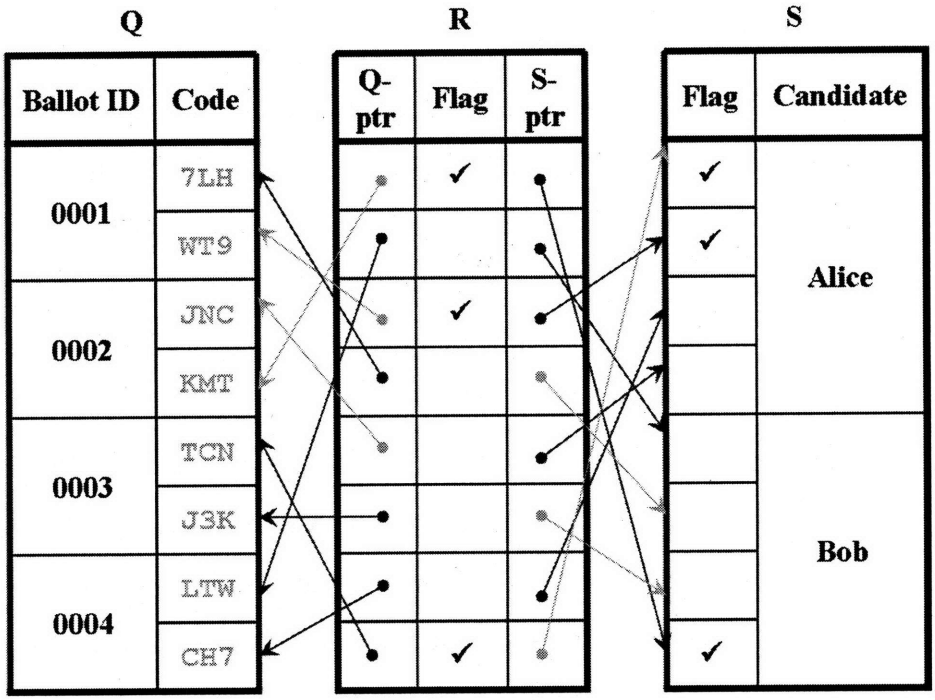

Figure 3-3: Tables **Q**, **R**, and **S** published by election officials after the close of the election. Entries shown in gray remain web-hidden. Those shown in black are web-revealed. For each web-revealed code in **Q** (except those corresponding to ballot #0004, an audited ballot), the corresponding rows in **R** and **S** have been flagged. For each row in **R**, either the **Q**-pointer or the **S**-pointer (or both, in the case of an audited ballot) has been web-revealed. The tally can be computed by counting the number of flags in each candidate block.

If any of the web-revealed codes differs from the codes on a voter's receipt, the voter may file a dispute.

3.9 Filing Disputes

We will refer to a person filing a dispute as a claimant. To file a dispute, a claimant must be on the voter registration list. The claimant submits a sheet of paper (which may or may not be the original receipt detached from the ballot) containing the ballot ID, the race in dispute, and a claim of a web-hidden code for that race. For each dispute filed, a record is made of the name of the claimant and the race in dispute, so that each voter may file disputes on at most one ballot ID and at most one per race. Note that multiple people may file disputes on the same ballot ID.

3.10 Handling Disputes

There are five circumstances under which disputes may arise:

1. The voter made a transcription error when recording the paper-revealed codes from the ballot.

2. The web-revealed code matches the paper-revealed code, but the voter is attempting to disrupt the election by submitting a random guess of a web-hidden code for the ballot.

3. A paper-revealed code has been web-hidden due to scanning error, and the voter is submitting this code.

4. A paper-revealed code has been web-hidden due to fraudulent behavior, and the voter is submitting this code.

5. A paper-revealed code has been web-hidden due to fraudulent behavior, and the claimant (presumably someone other than the voter of the ballot, in collusion with the adversary) is attempting to cover up the fraud by submitting a random

guess of a web-hidden code or a code which is known not to be a web-hidden code for the ballot.

After the end of the period for filing disputes, election officials reveal all of the web codes. Disputes containing web-hidden codes are called "plausible disputes". Plausible disputes are strong evidence of scanning error or fraud, since the probability of a voter correctly guessing a web-hidden code if it was not paper-revealed is very low. Plausible disputes are identified and counted.

We would like to distinguish between disputes resulting from scanning error or fraud (cases 3 and 4) and those resulting from transcription errors or random or incorrect guessing (cases 1, 2, and 5). Moreover, we would like to be able to distinguish between fraud and scanning error in the case that the amount of fraud committed is significant enough that the resulting discrepancies cannot be reasonably be attributed to scanning error. To do this, we can set a statistical trigger (described in section 4.1.4) based on the election parameters, such that if the number of plausible disputes exceeds a given threshold, the trigger is tripped, indicating strong evidence of fraud, and the election outcome should be declared invalid.

3.11 Checking the Tally

As described in section 3.7, for each element in **R**, election officials are asked to web-reveal either the **Q**-pointer or the **S**-pointer, depending on a pseudorandom, publicly verifiable coin flip.

To check the tally, any interested party can check the following properties:

- The commitments to the pointers are correct.
- Each web-revealed **Q**-pointer in table **R** either connects a web-revealed code in table **Q** to a flagged element in table **R** or connects a web-hidden code in table **Q** to an unflagged element in table **R**.
- Each web-revealed **S**-pointer in table **R** either connects a flagged element in

table **R** to a flagged element in table **S** or connects an unflagged element in table **R** to an unflagged element in table **S**.

Essentially, the audit checks that flags are mapped unchanged from table **Q** through table **R** to table **S**, where they are used to compute the tally.

Finally, any interested party can check that the tally for each candidate is correctly computed as the total number of flagged entries in that candidate's block in table **S**.

Chapter 4

Security Analysis

We now analyze the security properties of Scantegrity II. We consider separately the integrity and the privacy of the system.

We show that Scantegrity II provides greatly improved integrity compared to conventional optical scan voting. However, Scantegrity II is vulnerable to the threat that an adversary could modify voted ballots to appear as though they were spoiled for audit, effectively deleting those ballots. The voters of those ballots would be able to detect the malfeasance but would not be able to provide proof of it to election officials. We propose a modification to Scantegrity II to address this threat.

Although Scantegrity II provides increased assurance of integrity, the additional information printed on ballots makes the voter privacy of Scantegrity II more fragile than that of conventional optical scan voting. There are several potential threats to voter privacy, but we consider these to be manageable through careful control of access to ballot printing information and ballot boxes.

In our analysis, we consider possible adversaries to be election officials, poll workers, voters, and external parties.

4.1 Integrity

Integrity refers to the inability of an adversary to modify votes undetectably. Specifically, we are interested in detecting when votes are not correctly counted as cast

(because they are either not collected as cast or not counted as collected), even if the level of fraud does not actually change the winner of the election.

At a high level, the integrity of the system is provided through voter verification that votes are collected as cast and universal verification that all votes are counted as collected. During the voting process, the voter may copy down the confirmation codes paper-revealed by her choices. Later, when she checks her receipt against the bulletin board, if the paper-revealed codes match the web-revealed codes, the voter gains assurance that a vote for her chosen candidate was collected, since the auditing of a random selection of ballots ensures that ballots are printed correctly with respect to the bulletin board commitments. Finally, anyone can verify the correctness of the tally with respect to the collected votes by (1) checking the pointers revealed via randomized partial checking and (2) summing the number of flags corresponding to each candidate. Modifying the tally would require flipping the flags in at least one of the tables, and each flip has a 1/2 chance of being detected.

We now state our assumptions and consider various threats in detail.

4.1.1 Assumptions

We make the following assumptions for the integrity of the system.

- Some sufficient fraction of the printed ballots available for voting is randomly selected to be audited for correctness and consistency with the bulletin board commitments. Before the pre-election audit phase, an adversary has no non-negligible advantage in guessing which printed ballots will be audited.

- Some minimally sufficient number of voters check their receipts against the bulletin board. Before the post-election bulletin board posting, an adversary has no non-negligible advantage in guessing which receipts will be checked.

- All voters vote using a decoder pen (not a regular pen), paper-revealing the codes associated with their choices.

- All voters use a "none of the above" option to submit any undervotes.

- All printed ballots available for voting are either voted or audited. (We will also consider the effect of relaxing this assumption.)

4.1.2 Ballot Audit

Generating Repeated Confirmation Codes

As stated in section 3.3, confirmation codes are required to be unique within each race on a ballot. If confirmation codes are improperly generated so that there are repeated codes within a single race on a ballot, then voter verification that the web-revealed code matches the paper-revealed code does not ensure that a vote was collected as cast, since the web-revealed code may correspond to a candidate other than the one chosen by the voter. The uniqueness of codes within each race is checked during the ballot audit. Letting B be the total number of ballots and A be the number of audited ballots, the probability of an adversary printing k ballots with repeated confirmation codes without being detected is $\binom{(B-k)}{A}/\binom{B}{A}$, which can be shown to be upper bounded by

$$\min[(1 - A/B)^k, (1 - k/B)^A]. \tag{4.1}$$

As an example, if $B = 2000$ ballots are printed, $A = 1000$ ballots are audited, and $k = 20$ (2% of the voted ballots) are "bad", the probability of all of the bad ballots going undetected is less than 10^{-6}.

Printing Ballots Improperly

Ballots may be printed improperly in the following ways:

- The paper codes on a printed ballot are the same as the web codes for that ballot, but the codes are printed in a different order from that specified by the entries committed to in the tables. If the paper codes are not printed in the correct order, then voter verification that a web-revealed code matches a paper-revealed code does not ensure that a vote was collected as cast, since the web codes have a different correspondence to the candidates than the paper codes.

- The paper codes do not match the web codes, in any order. This could be an attempt to eliminate a voter's ability to file a plausible dispute, since the voter's paper-revealed code would not match one of the web-hidden codes.

- The ballot ID on the voting portion does not match the ballot ID on the receipt portion, or the ballot ID is not among those published in table **Q**. This could also be an attempt to eliminate a voter's ability to file a plausible dispute.

Any of these methods of printing ballots improperly will be detected by the ballot audit with probability upper bounded by Equation (4.1).

Modifying Paper Ballots

After the printing of the ballots, an adversary may try to modify the ballots, either by tampering with the invisible ink confirmation codes or by substituting fraudulent ballots for the printed ones. Under the assumption that the ballots randomly selected to be audited are selected from the same pool as those given out for voting, this will be caught by the ballot audit, again with probability upper bounded by Equation (4.1).

4.1.3 Voter Verification

Adding or Deleting Web Ballots

Adding ballots to the bulletin board (but leaving the cast ballots unchanged) would require adding names to the published list of voters who voted and changing the number of voters reported by the precincts. If an adversary adds a significant number of names to the list of voters, this should be detected by someone (someone might know that a person on the list did not actually vote).

An adversary could attempt to delete a voted ballot by marking it as spoiled for audit and web-revealing all of the codes. The voter of the ballot can detect this if and when she checks her receipt against the bulletin board. However, the voter cannot file a plausible dispute since she does not have a web-hidden code (all of the previously

web-hidden codes have been web-revealed). From the election officials' perspective, she might have spoiled the ballot for audit and then claimed that she voted the ballot.

However, assuming that all ballots are either voted or audited, and that an adversary cannot change the total number of voted ballots undetectably, if an adversary marks a voted ballot as audited, he will also need to change an audited ballot to a voted ballot. The latter can be detected by the voter checking the audited ballot that was changed to a voted ballot, and the voter can provide web-hidden codes to file a plausible dispute.

The assumption that all ballots that are not voted are audited may not be realistic. Furthermore, one might worry that some voters checking the audited ballots might be in collusion with the adversary, so we cannot rely on voters filing disputes for audited ballots changed to voted ballots in order to detect voted ballots being changed to audited ballots.

We propose a modification to address the threat of an adversary changing voted ballots into audited ones. The basic idea is that the voter voting on a ballot should obtain some information (e.g., confirmation code) that a voter auditing the ballot does not. We propose adding two confirmation codes called "vote codes", printed in invisible ink, to each ballot. To vote a ballot, the voter must paper-reveal both vote codes. When auditing a ballot, the voter chooses one of the vote codes to paper-reveal; the other vote code is detached from the ballot and destroyed. Thus, a voter of a ballot that has been incorrectly marked online as audited can provide the vote codes to file a plausible dispute.

Another possible method of deleting voted ballots would be to print multiple copies of a single ballot, so that multiple voters might check the same ballot ID and be falsely assured that their votes were collected as cast, when only one copy of their votes was collected. This method runs of the risk of different voters of duplicated ballots voting differently. Since only one set of codes is web-revealed for a single ballot ID, some voters will have paper-revealed codes that are web-hidden and will be able to file plausible disputes.

Adding Marks to Ballots

Assuming that voters use a "none of the above" option to submit undervotes, a voter can always copy down a code for each race, whether or not she selects a candidate. Therefore, if an adversary adds a mark to a ballot and web-reveals an additional code, the ballot becomes overvoted (and should not have been accepted by the optical scanner).

Web-Revealing Incorrect Codes

If an adversary web-reveals a code other than the one corresponding to the voter's candidate choice, he risks the voter checking her receipt against the bulletin board and filing a plausible dispute.

4.1.4 Statistical Triggers

To handle disputes arising from the voter verification process, we set up a statistical trigger such that if the number of plausible disputes exceeds some threshold, the trigger is tripped, indicating strong evidence of fraud, and the election outcome should be declared invalid.

Here we consider one possible statistical trigger. Consider a single disputed race. Let N be the number of candidates, C be the code space, D be the total number of disputes filed, and W be the number of plausible disputes filed.

No Scanning Errors

We first consider the ideal case, in which there are no scanning errors. If all filed disputes are random guesses, then the distribution of W given D total disputes is the binomial distribution

$$P[W = w|D] = \binom{D}{w} \cdot p^w \cdot (1-p)^{(D-w)} \tag{4.2}$$

where $p = (N-1)/(C-1)$ is the probability of a single guess of a web-hidden code being correct.

We set the trigger τ such that the probability of the number of plausible disputes being at least τ, if all filed disputes are random guesses, is less than 1%. We use the following bound on the right tail of the binomial [10]. For $r > \mu$,

$$P[W - \mu \geq r] \leq \left(\frac{\mu e}{r}\right)^r \tag{4.3}$$

where $\mu = Dp$ is the expected number of correct guesses of web-hidden codes.

Using example numbers of $N = 5$ candidates, $C = 20^3 = 8000$ possible codes, and $D = 100$ disputes filed, $p = 4/7999 = 0.0005$, and we get

$$P[W \geq 0.05 + r] \leq \left(\frac{0.05e}{r}\right)^r. \tag{4.4}$$

Setting r to 1.95, we get

$$P[W \geq 2] \leq 0.0055 \leq 0.01. \tag{4.5}$$

Therefore, if at least 2 out of 100 disputes were plausible disputes, the trigger would be tripped.

Scanning Errors

We now consider how to detect fraud in the more realistic case that scanning errors are possible. Our solution is to use past data on the scanners and federal certification standards to estimate an upper bound s on the rate of scanning error. We incorporate s into the probability p of a single claimed code being a web-hidden code.

If all filed disputes are either the result of scanning errors or random guesses, then the distribution of W given D total disputes is

$$P[W = w|D] = \binom{D}{w} \cdot p^w \cdot (1-p)^{(D-w)} \tag{4.6}$$

where $p = (N-1)/(C-1) + s$ is the probability of a single claimed code being a web-hidden code.

The statistical trigger can be computed as above, using the new value of p. Using example numbers of $N = 5$ candidates, $C = 20^3 = 8000$ possible codes, $D = 100$ disputes filed, and a maximum acceptable scanning error rate of $s = 0.001$, we get $p = 4/7999 = 0.0015$, and

$$P[W \geq 0.15 + r] \leq \left(\frac{0.15e}{r}\right)^r. \tag{4.7}$$

Setting $r = 2.85$ we have

$$P[W \geq 3] \leq 0.0039 \leq 0.01. \tag{4.8}$$

Therefore, if at least 3 out of 100 disputes were plausible disputes, the trigger would be tripped.

4.1.5 Universal Verification

Modifying the Tally

Assuming that an adversary cannot modify the web-revealed confirmation codes without detection, an adversary trying to modify the tally would have to flip some flags in table **S**. For each flag in table **S** that the adversary changes, he can either change the flag pointing to it in table **R** or not. If the adversary does not flip the flag in table **R**, he has a 1/2 chance of being detected because opening the **S**-pointer will reveal that the flags in table **R** and in table **S** are different. Similarly, if the adversary flips the flag in table **R**, he has a 1/2 chance of being detected if the **Q**-pointer is opened. Thus, the probability of an adversary changing k flags without being detected is $1/2^k$.

4.2 Privacy

Privacy refers to the inability of an adversary to associate a voter's candidate choice with the voter. Assuming that voters do not make distinguishing marks on their ballots, conventional optical scan voting systems provide information-theoretic privacy for voters. That is, a voter's ballot is hidden with equal probability among all of the cast ballots. In Scantegrity II, because ballots are printed with ballot ID's and different patterns of confirmation codes, we do not achieve information-theoretic privacy. Given a voter's ballot ID and full access to the ballot boxes, an adversary could determine how the voter voted. In order to manage this threat, we rely on carefully controlled access to the ballot boxes, as well as to the table $\mathbf{P}$ of confirmation codes sent to the printer.

We now state our assumptions and discuss the privacy of the system in more detail.

4.2.1 Assumptions

We make the following assumptions for the privacy of the system.

- Confirmation codes, $\mathbf{Q}$- and $\mathbf{S}$-pointers, and randomization used for bulletin board commitments are pseudorandomly generated by election officials using a trusted workstation.

- There are no recording devices (e.g., cameras) in the polling booth.

- No record is made by poll workers of the ballot ID's of ballots given to individual voters.

- The invisible ink confirmation codes cannot be read before being revealed by a decoder pen.

4.2.2 Confirmation Codes

Since the confirmation codes are pseudorandomly generated and pseudorandomly assigned to candidates, an adversary does not learn any information about a voter's candidate choice from the web-revealed or paper-revealed code for that candidate. If the web-revealed code does not match the paper-revealed code, an adversary still does not learn any information about a voter's candidate choice.

4.2.3 Randomized Partial Checking

For each code, either the **Q**-pointer or the **S**-pointer is revealed. Suppose the **Q**-pointer for a particular code is revealed. Then the index of the code in table **S** is one of those not pointed to by a revealed **S**-pointer. Similarly, if the **Q**-pointer for a code is not revealed, then the **S**-pointer is revealed, so the index of the code in table **S** is one of those pointed to by a revealed **S**-pointer. Thus, the randomized partial checking of the pointers hides each voter's vote equally among a randomly selected half of the votes.

4.2.4 Access to Ballot Box or Table P

If an adversary had access to the ballot box and a voter's ballot ID, he could find the voter's ballot and determine how she voted. Therefore, access to the ballot box must be carefully controlled.

Similarly, if an adversary had access to the confirmation codes in table **P** and a voter's receipt, he could determine how she voted. The voter would not be able to give a fake receipt to an adversary with access to table **P** because she would not be able to guess the other codes on that ballot or codes on other ballots. Therefore, election officials and poll workers must ensure that table **P** is only generated for the printers and later re-generated for the post-election posting, and table **P** is never saved or distributed.

4.2.5 Changing a Voted Ballot to an Audited Ballot

As discussed in section 4.1.3, an adversary might change some voted ballots to audited ballots on the bulletin board. The proposed modification of adding two vote codes to the ballot allows the voters of these ballots to file plausible disputes. However, since the correspondence between confirmation codes and candidates for these ballots has already been revealed on the bulletin board, an adversary with access to a voter's receipt would be able to determine how she voted. Therefore, voters should be encouraged to keep their receipts private, especially before the post-election posting.

4.3 Incoercibility

We have argued that Scantegrity II protects voter privacy, i.e., an adversary cannot determine how an individual voter voted. However, privacy does not imply incoercibility. In this section we give a brief argument that Scantegrity II is not vulnerable to coercion attacks.

Incoercibility refers to the inability of an adversary to coerce or provide incentive for voters to vote a particular way, including for a particular candidate, against a particular candidate, for a random candidate, and for no candidate. A coercion strategy may even be probabilistic; for example, a coercer may be able to coerce voters who receive certain ballots but not others.

Under the same assumptions as those for the privacy of the system, Scantegrity II is not vulnerable to coercion attacks because, before being marked, the ballot contains no visible information beyond what appears on conventional optical scan ballots (i.e., race titles and candidate names) except the ballot ID. In particular, all of the confirmation codes are hidden to the voter, so the voter cannot be forced to reveal a confirmation code satisfying any particular property. Furthermore, any revealed confirmation code could equally plausibly correspond to any of the candidates. Therefore, a confirmation code on a voter's receipt does not give any information about the candidate choice or the index of the candidate choice.

Chapter 5

Conclusion

In this thesis we surveyed several paper-based end-to-end verifiable voting systems and discussed various coercion attacks against some of them, illustrating the notion that incoercibility is a stronger security requirement than voter privacy.

We described the Scantegrity II system, an end-to-end verifiability mechanism for optical scan voting using confirmation codes printed on ballots using invisible ink. Scantegrity II allows voters to verify that their votes are collected as cast and allows anyone to verify that votes are counted as collected. Scantegrity II offers several advantages over existing end-to-end voting systems, including compatibility with existing optical scan technology, a mostly familiar voting experience, security against randomization coercion attacks, and a clean mechanism for handling disputes.

We analyzed the verifiable integrity and privacy properties of Scantegrity II. The system provides a high level of integrity, although the threat of voted ballots being changed into audited ballots needs to be addressed. We proposed a modification which addresses this threat but would make the system slightly more complicated and would affect the computation of statistical triggers for handling disputes. However, the ballot ID's and confirmation codes used for voter verification make the privacy of the system more fragile than that of conventional optical scan voting. Therefore, care must be taken to ensure proper handling and access control of the confirmation codes sent to the printer and the physical ballot boxes.

Scantegrity II represents significant progress in the design of usable and practical

end-to-end verifiable voting systems. However, in order for voting systems such as Scantegrity II to be adopted and accepted, much work remains to be done. Research needs to be done to more fully address the privacy issues of Scantegrity II, conduct full-scale usability studies, and ensure that Scantegrity II is compliant with various election laws. The challenge is to bring Scantegrity II to the point that it can be widely used to provide high integrity to large-scale elections.

Bibliography

[1] Voting system.

[2] Ben Adida. *Advances in Cryptographic Voting Systems.* PhD thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, August 2006.

[3] Ben Adida and Ronald L. Rivest. Scratch & vote: self-contained paper-based cryptographic voting. In *WPES '06: Proceedings of the 5th ACM workshop on Privacy in electronic society*, pages 29–40, New York, NY, USA, 2006. ACM Press.

[4] G.R. Blakley. Safeguarding cryptographic keys. *Proceedings of the National Computer Conference*, 48:313–317, June 1979.

[5] Jens-Matthias Bohli, Jörn Müller-Quade, and Stefan Röhrich. Bingo voting: Secure and coercion-free voting using a trusted random number generator. In Ammar Alkassar and Melanie Volkamer, editors, *Proceedings of the First Conference on E-Voting and Identity (VOTE-ID)*, volume 4896 of *LNCS*, pages 111–124. Springer, 2007.

[6] David Chaum. Secret-ballot receipts: True voter-verifiable elections. *IEEE Security and Privacy*, 02(1):38–47, 2004.

[7] David Chaum, Richard Carback, Jeremy Clark, Aleksander 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. To appear at USENIX/ACCURATE Electronic Voting Technology Workshop (EVT '08).

[8] 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*, May/June 2008.

[9] David Chaum, Peter Y.A. Ryan, and Steve A. Schneider. A Practical, Voter-verifiable, Election Scheme. Technical Report Series CS-TR-880, University of Newcastle Upon Tyne, School of Computer Science, December 2004.

[10] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms.* The MIT Press and Mc-Graw Hill, second edition, 2001.

[11] Aleks Essex, Jeremy Clark, Richard Carback, and Stefan Popoveniuc. The Punchscan voting system: VoComp competition submission. In *Proceedings of the First University Voting Systems Competition (VoComp)*, 2007.

[12] M. I. Finley. *Politics in the Ancient World.* Cambridge University Press, 1983.

[13] Andrew Gumbel. *Steal This Vote: Dirty Elections and the Rotten History of Democracy in America.* Nation Books, 2005.

[14] Markus Jakobsson, Ari Juels, and Ronald L. Rivest. Making mix nets robust for electronic voting by randomized partial checking. In Dan Boneh, editor, *Proceedings of the 11th USENIX Security Symposium*, pages 339–353. USENIX Association, 2002.

[15] Douglas W. Jones. The evaluation of voting technology. In Dimitris A. Gritzalis, editor, *Secure Electronig Voting.* Kluwer Academic Publishers, 2003.

[16] John Kelsey, Andrew Regenscheid, Tal Moran, and David Chaum. Untitled paper on coercion attacks on E2E systems. Currently not submitted for publication.

[17] Rebecca Mercuri. *Electronic Vote Tabulation Checks and Balances.* PhD thesis, University of Pennsylvania, Philadelphia, PA., October 2000.

[18] Tal Moran and Moni Naor. Split-ballot voting: Everlasting privacy with distributed trust. In *CCS 2007*, pages 246–255. ACM, 2007.

[19] Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *CRYPTO '91: Proceedings of the 11th Annual International Cryptology Conference on Advances in Cryptology*, pages 129–140, London, UK, 1992. Springer-Verlag.

[20] Stefan Popoveniuc, Jeremy Clark, Aleks Essex, Richard T. Carback III, and David Chaum. Securing optical-scan voting. In *Proceedings of Frontiers of Electronic Voting*, LNCS. Springer. To be published in 2008.

[21] Stefan Popoveniuc and Ben Hosp. An introduction to Punchscan. In *Preproceedings of the 2006 IAVoSS Workshop on Trustworthy Elections (WOTE)*, Robinson College, Cambridge, United Kingdom, 2006. International Association for Voting System Sciences.

[22] Caltech/MIT Voting Technology Project. *Voting: What Is, What Could Be.* July 2001.

[23] Ronald L. Rivest and Warren D. Smith. Three voting protocols: Threeballot, VAV, and Twin. In *USENIX/ACCURATE Electronic Voting Technology Workshop (EVT)*, August 2007.

[24] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.