Election Security is Harder Than You Think

by

Matthew Bernhard

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Computer Science and Engineering)
in the University of Michigan
2020

Doctoral Committee:

Professor J. Alex Halderman, Chair
Assistant Professor Nikola Banovic
Research Professor Peter Honeyman
Professor Walter R. Mebane, Jr.
Institute Professor Ronald L. Rivest

Matthew Bernhard

matber@umich.edu

ORCID iD: 0000-0002-2700-8921

DEDICATION

To Mom and Dad, who raised me to be thoughtful, compassionate, and curious.

ACKNOWLEDGEMENTS

Thanks to my parents for supporting my going to grad school even though it wasn't always clear why. Thanks to my mom for supporting me no matter what. I regret that you'll never get to read this. Thanks to my dad for being a shelter from the storm and a shoulder to lean on throughout the very difficult past year, and for helping me throughout life in all manner of things.

Thanks to Monica, without whom the last mile would have been unfinished.

Thanks to Ben VanderSloot, who has been in the trenches with me since day one, and who has taught me so much about how to engage critically in the world. Thanks to Allison McDonald, for being a best friend when I needed one most, and for helping me recalibrate my understanding and expectations of the world. Thanks to Ram Sundara Raman for comisserating and listening. Thanks to Reethika Ramesh for giving me hope that good things and good people can survive grad school.

Thanks to Sai Gouravajhala, David Adrian, Zakir Durumeric, Pat Pannuto, Eric Wustrow, Drew Springall, Andrew Kwong, Kevin Loughlan, Andrew Quinn, Chris Dzombak, Deepkika Natarajan, Marina Minkin, Haizhong Zheng, Tim Trippel, Ofir Weisse, Renuka Kumar, Steve Sprecher, Connor Bolton, and so many others who made grad school what it was.

Thanks to Dan Wallach, for putting me on this path to begin with. Thanks to J. Alex Halderman for taking a chance on an undergrad with a less-than-stellar GPA, and facillitating my growth into something approaching a scientist.

Thanks to my committee members, who have all been incredibly supportive.

TABLE OF CONTENTS

DEDICATION ii

ACKNOWLEDGEMENTS iii

LIST OF FIGURES ix

LIST OF TABLES xii

ABSTRACT xiii

CHAPTER

I. Introduction 1

1.1 What Makes an Election Secure? 2
1.1.1 Why Paper Matters 3
1.2 Voter-verified Paper 4
1.3 Post-election Audits 5
1.3.1 Modeling Post-election Audits 7
1.3.2 A Manifest Improvement on an Existing RLA Technique 7
1.4 Perfect is the enemy of good 8

II. What Makes an Election Secure? 10

2.1 Introduction: What is the evidence? 10
2.1.1 Why so hard? 11
2.2 Definitions of Integrity 12
2.2.1 Software Independence 13
2.2.2 End-to-end verifiability 14
2.2.3 From Verifiable to Verified 16
2.3 The Role of Paper and Ceremonies 17
2.3.1 Risk-Limiting Audits 18
2.4 Privacy, Receipt Freeness, and Coercion Resistance 20
2.4.1 Basic Confidentiality 20

2.4.2 Everlasting Privacy 21
2.4.3 Systemic Privacy Loss 21
2.4.4 Receipt-freeness 22
2.4.5 Coercion Resistance 23
2.5 Other Security Considerations 24
2.5.1 Voter Authentication 24
2.5.2 Availability 25
2.5.3 Usability 25
2.5.4 Local Regulatory Requirements 26
2.5.5 Complex Election Methods 26
2.6 A Look Ahead 27

III. Why Paper Matters 28

3.1 Introduction 28
3.2 Background 30
3.2.1 Ballot Images 30
3.2.2 Image Audits 31
3.3 Attack Scenarios 32
3.4 Methodology 34
3.4.1 Reading the ballot 35
3.4.2 Changing marks 35
3.4.3 UnclearBallot 37
3.5 Evaluation 38
3.5.1 Testing Across Ballot Styles 39
3.5.2 Testing with Real Voted Ballots 41
3.6 Discussion and Mitigations 43
3.6.1 Uses for image audits. 44
3.6.2 End-to-end (E2E) systems. 44
3.6.3 Other mitigations. 45
3.6.4 Detection. 45
3.6.5 Future work. 46
3.7 Conclusion 47

IV. Voter-verified Paper 48

4.1 Introduction 48
4.2 Background and Related Work 51
4.2.1 Human-Dependent Security 51
4.2.2 Usable Voting Systems 53
4.2.3 Voter-Verifiable Paper and Ballot-Marking Devices 54
4.3 Materials and Methods 55
4.3.1 The Polling Place 57
4.3.2 The Voting Machines 58
4.3.3 The Ballot 59

4.3.4 Participants and Recruitment 61
4.3.5 Experiments 62
4.4 Results 64
4.4.1 Participant Demographics 64
4.4.2 Verification Performance 65
4.4.3 Correlates 67
4.4.4 Participant Comments 72
4.5 Security Model 74
4.6 Discussion 78
4.6.1 Limitations 78
4.6.2 Discussion of Findings 79
4.6.3 Recommendations 80
4.7 Conclusion 85

V. Modeling Post-election Audits 86

5.1 Introduction 86
5.2 What is a risk-limiting audit? 89
5.2.1 Audits in Complex Elections 90
5.2.2 Audit Units 91
5.2.3 A general RLA 91
5.2.4 Mechanics of an RLA 92
5.3 Workload Estimation 94
5.3.1 Preliminaries 94
5.3.2 The Workload Function 97
5.3.3 Estimating Workload 98
5.3.4 An example workload comparison 100
5.4 Parallelizing the work 100
5.5 Validating the Model 101
5.6 Comparing RLA Techniques 103
5.6.1 Election Size 105
5.6.2 Batch Size 106
5.6.3 Escalation 107
5.7 Limitations 108
5.7.1 Future Work 109
5.8 Conclusion 109

VI. A Manifest Improvement on an Existing RLA Technique 110

6.1 Introduction 110
6.2 Notation and Mathematical Background 114
6.2.1 Multi-round Bernoulli Sampling 114
6.2.2 Exchangeability and Conditional Simple Random Sampling . . 115
6.3 Tests 115
6.3.1 Wald's SPRT with a Nuisance Parameter 116

6.3.2 Auditing Multiple Contests 117
6.4 Escalation 118
6.5 Initial Sampling Rate 119
6.6 Implementation 121
6.6.1 Election Night Auditing 121
6.6.2 Vote-by-mail and Provisional Ballots 122
6.6.3 Geometric Skipping 122
6.6.4 Pseudorandom Number Generation 123
6.7 Evaluation 124
6.7.1 Empirical Data 125
6.8 Discussion 125
6.8.1 Previous Work 127
6.8.2 Stratified Audits 127
6.9 Conclusion 127

VII. Examining Coercion 129

7.1 Introduction 129
7.2 Modelling Vote-by-Mail 131
7.2.1 Functions 132
7.2.2 Setup 133
7.2.3 Assumptions 134
7.2.4 Examining VBM 135
7.2.5 Realistic Coercion Mitigation 137
7.3 Improving VBM 138
7.3.1 Improving Authentication 139
7.3.2 Improving the Voting Channel 139
7.3.3 VBM Improved 140
7.4 Discussion 142
7.4.1 Related Work 142
7.4.2 Usability 144
7.5 Conclusion 145

VIII. Conclusion 146

8.1 Future Work 148
8.2 Looking Ahead 149

Appendices 151

A. Voter-verified Paper 152

1.1 Poll Worker Script 152
1.2 Recruitment Script 153
1.3 Slate of Candidates for Directed Voting Condition 154

BIBLIOGRAPHY 155

LIST OF FIGURES

Figure

3.1 **Attack overview** — A voter's paper ballot is scanned by a ballot tabulator, producing a digital image. Malware in the tabulator—in our proof-of-concept, a microdriver that wraps the scanner device driver—alters the ballot image before it is counted or stored. A digital audit shows only the manipulated image. 29

3.2 **Terms for parts of a marked ballot**, following Jones [124]. 31

3.3 **Taxonomy of voter marks** adapted from Bajcsy [20], including the five leftmost marks that may be considered marginal marks. 32

3.4 **Ballot manipulation algorithm** — First, (a) we apply template matching to extract the race we intend to alter. Then, (b) we use Hough line transforms to separate each candidate. If the first candidate has a race title box, (c) we remove it by computing the pixel intensity differences across a straight line swept vertically from the bottom. For each candidate, (d) we identify the target and mark (if present) by doing four linear sweeps and taking pixel intensity. Finally, (e) we identify and move the mark. At each step we apply tests to detect and skip ballots where the algorithm might leave artifacts. 36

3.5 **Automatically moving voter marks** — UnclearBallot seamlessly moves marks to the attacker's preferred candidate while preserving the voter's marking style. It is effective for a wide variety of marks and ballot designs. In the examples above, original ballot scans are shown on the left and manipulated images on the right. . 37

3.6 The **Fujitsu fi-7180 scanner** we used to test our attack has been certified by the U.S. Election Assistance Commission for use in voting systems. Our proof-of-concept implementation is a malicious scanner driver that alters ballots on the fly. 38

3.7 **Ballots Styles** — We tested ballot designs from five U.S. voting system vendors: Clear Ballot, Diebold, Dominion, ES&S, and Hart (two styles, eScan and Verity). 40

3.8 **Attacking Real Ballots** — Using 181,541 images of voted ballots from Clackamas County, Oregon, we attempted to change voters' selections for the ballot measure shown above. UnclearBallot determined that it could safely alter 34% of the ballots. For reference, Measure 102 passed by a margin of 5%, well within range of manipulation [70]. We inspected 1,000 of them to verify that the manipulation left no obvious artifacts. 42

4.1 *Polling Place Setup.* We established mock polling places at two public libraries in Ann Arbor, Michigan, with three BMDs (*left*) and an optical scanner and ballot box (*right*). Library visitors were invited to participate in a study about a new kind of election technology. The BMDs were DRE voting machines that we modified to function as malicious ballot marking devices. 56
4.2 *Ballot Styles.* We tested two ballot styles: (*a*) a regular style, resembling a hand-marked ballot; and (*b*) a summary style, listing only the selected candidates. Both had 13 races from the city's recent midterm election. In one race, determined randomly, the printed selection differed from the voter's choice. 60
4.3 *Warning Signage.* One of the interventions we tested was placing a sign above the scanner that instructed voters to verify their ballots. Signage was not an effective intervention. 63
4.4 *Participant Demographics* Our participants largely reflected the demographics of Ann Arbor: they were well educated, mostly white, and mostly women. 65
4.5 *Ballot Submission Times for Different Instructions.* Histogram showing the time from ballot printing to ballot submission for two sets of experiments: ones where participants were given instructions designed to increase verification and ones where participants received standard instructions. Participants in the former group took longer: 83 s on average vs. 50 s for those who received no special instructions. Voters that received extra instructions but who were not given a slate took an average of 62 s. 69
4.6 *BMD security is highly sensitive to human performance.* Given a 0.5% margin of victory, we plot the percentage of voters who report a problem during the minimal outcome-changing attack as a function of the rate at which errors are detected and corrected. This model implies that using BMDs safely for all voters requires dramatically improved verification performance or very sensitive attack detection thresholds. 76
5.1 *Workload estimates for three types of RLA*—The workloads for ballot-polling, ballot comparison, and batch comparison RLAs with a risk-limit of 10%, 100,000 ballots cast, and 1,000 ballots per batch, by margin. The graphs on the left and the right show the linear and parallel workloads, respectively. The key differences are the scale, and that batch comparison is completely flat in parallelized audits, since all batches are treated equally. 104
5.2 *Crossover point for batch comparison and ballot polling*— Ballot-polling audits become less efficient than batch comparison audits when the margins dip below 2 to 3% in elections with 100,000 ballots cast and a batch size of 1,000, depending on the risk-limit as depicted here. 105
5.3 *The effect of election and batch size on efficiency*—The relative workloads for ballot-polling, ballot comparison, and batch comparison RLAs vary widely depending on the size of the election (shown here with a risk-limit of 10%). For smaller elections, batch comparison beats out ballot polling with relatively wide margins. The efficiency of ballot comparison audits scales well, however, and its sample sizes and therefore workloads are relatively small regardless of election size. Batch sizing also has essentially no effect on ballot-polling and -comparison audits, but smaller batch sizes do make batch compraison more efficient. 106

5.4 *Escalation of a ballot-polling RLA*—Escalation can cause the workload for ballot-polling to explode rather quickly, even in elections with wide margins. Here we depict the additional workload of escalating a sample if the first round sample shows a tie between the winner and the runner up, in an election with 100,000 ballots and a batch-size of 1,000. Interestingly, even for elections with wider margins, it becomes much more efficient to switch to a batch-comparison audit rather than continue ballot polling. 107
6.1 **Bernoulli ballot-polling audit step-by-step procedures.** 113
6.2 **Simulated quantiles of sample sizes by fraction of votes for the winner for a two candidate race** in elections with 10,000 ballots and 1 million ballots, for BRAVO ballot-polling audits (BPA) and Bernoulli ballot polling audits (BBP), for various risk-limits. The simulations assume every ballot has a valid vote for one of the two candidates. 124
7.1 **Absentee Ballot Envelope from Washington County, Oregon**—An example of the envelope a voter signs when mailing in their ballot [88]. 131
7.2 **Comparison between VBM procedure and VBM-Improved procedure** 143

LIST OF TABLES

Table

3.1 **Performance of UnclearBallot** — We tested how accurately our software could manipulate voter marks for a variety of ballot styles using equal numbers of invalid and valid marks. The table shows how often the system skipped a mark, successfully altered one, or erroneously created artifacts we deemed to be visible upon manual inspection. We also report the mean processing time for successfully manipulated races, excluding template matching. 41

4.1 *Verification Performance for Each Experiment.* Without interventions, participants' verification performance was remarkably poor: only 7.8% noted on an exit survey that their ballots had been altered, and only 6.6% informed a poll worker (averaged across experiments). The various interventions we tested had widely different effects, ranging from no significant improvement (**E4**, **E5**) to a large increase in verification success (**E8**, **E9**). 66

4.2 *Participant Attentiveness.* Voters who noticed the discrepancy tended to vote in every race and ballot position more often than those who did not. 70

5.1 **Workload Calculations for a Ballot-Polling Audit, in Seconds**—This table shows the workload calculations for a ballot-polling audit with a 10% risk-limit using the countdown method. Since handling the ballots is the most expensive part of auditing in this way, the $f(b_i)$ term dominates the workload as the number of ballots increases. 101

5.2 **Parallelization of a Ballot-Polling RLA**—Here we show the workloads achievable if every batch can be opened and its ballots audited simultaneously. Speedup depends on the total number of batches (hence our speedup here approaching 100) as well as the number of available auditing teams. W_{10} shows the workload if ten audit boards are working simultaneously, achieving a speedup of around 10. . . . 101

5.3 **Data from December 2019 Pilots**—Here we show data from three ballot-polling pilots conducted in December 2019 in Rhode Island, Lansing, and St. Joseph County Michigan. The error in our model's workload prediction for each of these audits is relatively small, on the order of 20 minutes or so for each audit. This is accounted for by the fact that the model does not include setup times for the audit, lunch breaks, etc. 102

6.1 **Estimated sampling rates needed for Bernoulli ballot polling** for a 2-candidate race with a 5% risk limit. These simulations assume the reported margins were correct. 120

ABSTRACT

Recent years have seen the rise of nation-state interference in elections across the globe, making the ever-present need for more secure elections all the more dire. While certain common-sense approaches have been a typical response in the past, e.g. “don’t connect voting machines to the Internet” and “use a voting system with a paper trail”, known-good solutions to improving election security have languished in relative obscurity for decades. These techniques are only now finally being implemented at scale, and that implementation has brought the intricacies of sophisticated approaches to election security into full relief.

This dissertation argues that while approaches to improve election security like paper ballots and post-election audits seem straightforward, in reality there are significant practical barriers to sufficient implementation. Overcoming these barriers is a necessary condition for an election to be secure, and while doing so is possible, it requires significant refinement of existing techniques. In order to better understand how election security technology can be improved, I first develop what it means for an election to be secure. I then delve into experimental results regarding voter-verified paper, discussing the challenges presented by paper ballots as well as some strategies to improve the security they can deliver. I examine the post-election audit ecosystem and propose a manifest improvement to audit workload analysis through parallelization. Finally, I show that even when all of these conditions are met (as in a vote-by-mail scenario), there are still wrinkles that must be addressed for an election to be truly secure.

Thesis Statement. Understanding the real-world complexities of administering elections is necessary to create and implement practical defenses.

CHAPTER I

Introduction

Democratic elections and the peaceful transfers of power they facilitate are the bedrock on which many cultures and ways of life are built. In many countries, elections provide one of the few touchstones for citizens to improve their government, and in doing so, their lives and the lives of those around them. However, democracy is an ever-imperfect process, and effectively carrying out its machinations is a very difficult task, even without undue interventions. A critical piece of of the democratic process is the integrity of elections, which relies not just on active and enthusiastic participation of the populace, but in the integrity of the systems by which people participate. Typically, this means integrity of the ballot casting, collecting, and tabulation process.

The means by which this process can be protected are varied, and depend largely on the specifics of a given election system. Voice-vote systems have a different threat model than mark-on-paper or mark-on-computer systems, and so forth. As much of the world relies on secrecy of each individual ballot, typical voting systems take the form of voter-marked paper or computers, which attempt to preserve the anonymity of each voter while still collecting and counting votes as efficiently as possible.

However, the systems by which people vote around the world have been routinely scrutinized in regards to the integrity and anonymity they can actually provide. Voting systems in the Netherlands [105], Brazil [16], India [247], Australia [111], Estonia [211], Switzerland [110], Norway [103], Canada [61], and the United States [18, 48, 67, 155, 160], to name a few, have all been found to severely underdeliver on the promise of a secure and correct election. This has lead many experts the world over to analyze the problem of running a secure election, develop concepts for what doing so requires, and build systems that can facilitate doing so.

However, despite having a well-stocked toolbox of techniques and policies to run secure elections, we have only just begun to see these tools picked up and used in a real-world context. In this dissertation, I examine some of these tools more closely: verifiable paper ballots, and post-election audits. First, I develop what it means for an election to be secure and why these two technologies are critical to election security. Then, I examine voter-verified paper ballots relying on experimental data collected in a mock-election setting, finding that while voters do not default to actually verifying their paper ballots, there appear to be strategies that can greatly improve the proportion of voters who do. I construct a model of post-election audits, examining various types of audits that have been proposed and implemented over the years, what factors impact the efficacy and efficiency of a post-election audit, and comparing and contrasting the efficiency of types of audits. Finally, I discuss how even if all of these technologies are deployed perfectly, as in the vote-by-mail scheme I examine, the attack surface of elections is so broad that it is still possible to have an insecure election with paper ballots and audits.

1.1 What Makes an Election Secure?

In order to rigorously examine election security technologies, it is first important to develop key concepts in election security and what types of systems and policies implement those concepts. Election security as a field has been developing for nearly forty years, with some of the earliest concepts like rigorous post-election audits and verifiability maturing over that time.

The security problem for elections is a particularly wicked one. The primary goal for an election is to select the most preferred candidate or choice amongst the electorate, however the secondary (and almost as important) function is to convince the losing sides of an election that they really lost. Providing this kind of assurance on its face is not that difficult: if everyone who votes can be tied to their vote, then all voters are capable of making sure their choice was collected and counted in the outcome, including the losers.

However, democracies the world over rely on a second key property for their elections: anonymity. If voters can reveal how they voted, they might become subject to undue influence over how they should vote, being paid to vote a certain way or suffering acts of violence when they do not vote a certain way. This type of undue influence is **coercion**, and is one of the main concerns when constructing a secure election system.

One of the most significant developments in response to coercion was the secret ballot, wherein a voter enters a private environment and marks a piece of paper with their intended selection. Provided the voter does not mark the paper in a particularly identifiable way, once the ballot has been mixed in with all of the other ballots in the election, it is essentially impossible to tie any one ballot back to any one voter. This effectively defeats any possibility of coercion, thus making the election scheme **coercion resistant**.

However, the introduction of paper ballots surfaces several other problems in running secure elections: how should the paper be counted? How does a jurisdiction produce, organize, administer, and collect paper ballots? What do voters who are unable to independently mark paper ballots do? To address these concerns, computers were introduced in the 1960s as an integral part of the voting process, ranging from electronic scanners that can process ballots more quickly and accurately than human counting teams to vote-marking computers that can facilitate voters of all abilities to vote independently and anonymously. The advent of computerized voting has even lead to the adoption of fully paperless voting systems around the world, from the Netherlands to Brazil to India to the United States.

1.1.1 Why Paper Matters

Election outcomes are subject to influence from all participants: candidates, parties, voters, voting equipment vendors, election officials, and, perhaps most saliently in light of recent elections, outside adversaries. Because of this, election systems need a sophisticated way to provide strong evidence for their outcomes while also preserving anonymity. Essentially, elections that use computers need a way to guarantee that their outcomes have not been influenced by any of the actors involved.

With computerized voting, the most essential weakness is undetected manipulation of the election results by malicious software. I developed software that can achieve exactly this type of manipulation by intercepting scanned images of ballots before they reach the tabulation software to change the votes reflected on them. This attack works specifically on voter-marked paper ballots, but similar attacks have been demonstrated on other computerized voting equipment like direct-recording electronic voting machines [18, 48, 155]. Because software can introduce changes to the election results, this study shows why it is absolutely crucial to maintain a robust paper trail attesting to the will of the voters.

To date, voter-verified paper is the only known mechanism by which elections can provide strong assurance to their results while also preserving anonymity in an environment where none of the stakeholders are trusted. Such assurances can also provide the property of **software-independence**: malicious or erroneous changes introduced by software which cause the election outcome to be incorrect cannot go undetected provided the computer-produced results are checked against the paper. However, "voter-verified" has proven to be an underdeveloped definition until recently.

1.2 Voter-verified Paper

The key idea of voter-verified paper is that once a voter marks their ballots, they can examine the marks on the ballot prior to casting it to ensure that the choices they made are the ones they intended. This way, truly voter-verified paper provides the property of **cast-as-intended**, one of the three properties of ensuring **end-to-end verifiability** (E2E-V). E2E-V provides assurance that an election outcome is correct from end-to-end, from when the ballot leaves the hands of each voter up to when the election outcome is announced. Providing for the other two properties of E2E-V, **collected-as-cast** and **tallied as collected**, requires a bit more work on the part of the election system, but I defer that discussion to Section 1.3.

While in principle any paper-based voting system can provide the property of cast-as-intended, it is important to note that it fundamentally requires on actions by the voters to do so. If no voter checks their ballot, then the election cannot provide end-to-end verifiability, as the paper trail may have been manipulated without anyone noticing. This manipulation can take the form of misprinted hand-marked paper ballots, or malicious changes to the ballots printed off by computerized ballot marking devices (BMDs).

To better characterize whether voters actually verify paper ballots, I performed a study running a mock election on BMDs in a realistic election environment. Every BMD was maliciously coded to intentionally change the voter's selections in controlled ways. With this control, I could vary factors about the way the ballots were manipulated or how the election environment was set up to ascertain what factors impact voters' ability to detect and report errors.

Absent any interventions, the proportion of participants in the study who reported problems on their ballots was dishearteningly low: fewer than 7% of participants found and reported errors, and only 40% appeared to even examine their paper ballot. However, I designed several interventions intended to improve both the review rates and the detection rates, and by asking voters to check their ballots carefully against a provided list of candidates they were instructed to vote for, our review rate approached 100% and the detection rate reached 62% (as high as 86% in one condition).

Therefore, it appears that it appears that voter-verified paper can deliver on its promise of software-independence and cast-as-intended, however it requires sophistication in implementation and design. Voters do not check their ballots on their own, and significant effort needs to be made in system design and policy implementation to achieve acceptable rates of verification. To date, at least two states have adopted my recommendations for use with their BMD systems.

1.3 Post-election Audits

Voter-verified paper is at the time of writing a necessary condition for secure elections, but it is not sufficient. The complement to voter-verified paper is a robust paper trail. If the paper trail is well preserved, i.e. there is significant evidence that voter-verified ballots have been transported, counted, and stored in a manner that did not result in any of them being removed or any dubious ballots being added, then a voting system can provide assurance to (but cannot guarantee) the property of collected-as-cast. Furthermore, if the paper trail is then audited against the reported election results in a robust way, then the election attains the property of tallied-as-collected and therefore approaches end-to-end verifiability.

Providing evidence to both a robust paper trail and a verified outcome can be achieved with audits that are performed after the election. Process audits can attest that the paper trail has been properly preserved. Tabulation audits can provide assurance that the paper trail affirms the reported election outcome. The work I present here addresses tabulation audits.

Perhaps the most significant development in post-election tabulation audits is the **risk-limiting audit** (RLA). The key idea is that a sample of paper ballots is retabulated by hand and compared against the voting system's reported outcome. This can be done at a macroscopic level, examining batches of ballots, or at a microscopic level, examining individual ballots.

The three most prominent types of RLA are **batch comparison** audits, **ballot-polling** audits, and **ballot comparison** audits. Each of these audits draw samples based only on the margin reported in the election, and then hand-tabulate the physical paper ballots. A hypothesis test is used to evaluate how likely it is that the reported election result is correct based on the sample.

RLAs are not the only type, or even the most commonly-used type of post-election tabulation audit, however. Many jurisdictions in the United States rely on fixed-percentage audits, wherein the jurisdiction retabulates some proportion of their paper trail, typically at a precinct-level, but occasionally at a machine- or ballot-level. Not all of these retabulations are performed by hand, either; many jurisdictions run their ballots through the same scanners that they used to tabulate the ballots originally, while some use different scanners.

Fixed-percentage audits are not guaranteed to provide assurances of E2E-V or software-independence, as the sample sizes they choose may be too small to do so with any statistical confidence. However, if the sample size is larger than necessary to be risk-limiting and is drawn uniformly at random, it is possible that these types of audits can provide security assurances. Nevertheless, they are not generally recommended as a security technique, as there is no way to guarantee whether they can provide statistical power. Furthermore, an adversary manipulating vote totals can ensure that a fixed-percentage audit cannot provide statistical confidence by choosing a low margin.

Some jurisdictions do not even examine their paper trail (even fewer do not even have one), they simply rerun the program that tabulated the outcome to verify that it is correct. This type of "audit", as it does not rely on any external evidence, can never provide any property of software-independence or E2E-V, so I will disregard it.

1.3.1 Modeling Post-election Audits

A key problem with the myriad varieties of post-election audits is that it is difficult to compare them. One of the most frequently asked questions by election officials considering implementing audits is "how much will it cost?" While fixed-percentage audits will have fixed costs, RLAs fundamentally depend on the election results to determine how much work they will entail. Therefore, it is not strictly possible to guarantee a given workload ahead of time.

However, it is possible to compare these methods for a given "average" election outcome. I develop a model to do just that, examining which types of post-election audits are the most efficient in which scenarios. I find that while certain audit methods like ballot comparison will always be most efficient, batch comparison and ballot-polling have trade-offs as the margins get small. The model I develop is fully generalizable to any post-election audit, and I validate it using data gleaned from my participation in numerous RLA pilots.

1.3.2 A Manifest Improvement on an Existing RLA Technique

In developing this model, I also note that a significant feature of post-election audits is their parallelizability, that many batches or ballots can be audited all at the same time. To that end, I propose a reexamination of the most widely-adopted form of RLA, the ballot-polling audit, and show that it is entirely possible to perform robust ballot sampling *before* the election results are made available, which addresses workload concerns and ameliorates some of the challenges in maintaining a paper trail.

1.4 Perfect is the enemy of good

Even if an election system has voter-verified paper ballots and a robust post-election audit, there are still means by which its result can be made incorrect. To demonstrate this, I examine a typical vote-by-mail (VBM) scheme used in the United States. In principle, this system has voter-verified, hand-marked paper ballots, and I assume that risk-limiting audits are performed. However, because of inadequate authentication of voters, the election is vulnerable to fraud and voter coercion.

I introduce several practical improvements to the scheme that can be deployed with relatively minimal changes to the election system. More robust authentication virtually eliminates the potential for fraud. My proposed scheme does not provide perfect coercion-resistance, however I develop a new concept of **coercion-hardness**, where a scheme permits some coercion but not enough to change the election outcome.

Despite the existence of strong practical defenses, I show that there is no "perfect" solution to election security. Even good defenses that are easy to deploy have important trade-offs that must be weighed carefully. A system that is perfectly secure but unusable is not a secure system. A system that is perfectly usable but cannot provide voters assurance to its correctness is not a usable system. However, by focusing on the goal of delivering a system that preserves outcomes if not individual votes, many improvements can be made over the status quo.

Thesis Statement. Understanding the real-world complexities of administering elections is necessary to create and implement practical defenses.

Structure. I have demonstrated in this dissertation that while there are powerful strategies for achieving more secure elections, their implementation is more nuanced and requires more sophistication than obvious at first blush. In Chapter II, I establish some important definitions for election security as well as some technologies that have been designed to meet those definitions. In Chapter III, I provide a practical example of how violating some of the core assumptions of those technologies can result in insecure elections. In Chapter IV, I explore voter-verified paper and demonstrate that having paper alone does not guarantee that voters will check it, as well as providing some strategies to ensure that they do. In Chapter V I develop a generalized model of post-election audits that rely on voter-verified paper, examining which types of audits are most effective when. In Chapter VI I examine how one of the key lessons from my model can be applied in practice by parallelizing an existing risk-limiting audit. In Chaper VII, I show that even with perfect deployment of paper ballots and RLAs, voting systems still possess security holes, and that defending these systems is fundamentally an act of compromise. Finally, in Chapter VIII, I draw broader conclusions from the work presented here, as well as outline areas in which more knowledge is needed to better inform election security technologies.

CHAPTER II

What Makes an Election Secure?[1]

2.1 Introduction: What is the evidence?

It is not enough for an election to produce the correct outcome. The electorate must also be convinced that the announced result reflects the will of the people. And for a rational person to be convinced requires evidence. Modern technology—computer and communications systems—is fragile and vulnerable to programming errors and undetectable manipulation. No current system that relies on electronic technology alone to capture and tally votes can provide convincing evidence that election results are accurate without endangering or sacrificing the anonymity of votes. Moreover, the systems that come closest are not readily usable by a typical voter.

Paper ballots, on the other hand, have some very helpful security properties: they are readable (and countable, and re-countable) by humans; they are relatively durable; and they are tamper-evident. Votes cast on paper can be counted using electronic technology; then the accuracy of the count can be checked manually to ensure that the technology functioned adequately well. Statistical methods allow the accuracy of the count to be assessed by examining only a fraction of the ballots manually, often a very small fraction. If there is also convincing evidence that the collection of ballots has been conserved (no ballots added, lost, or modified) then this combination—voter-verifiable paper ballots, a mechanized count, and a manual check of the accuracy of that count—can

[1] This chapter is based on “Public Evidence from Secret Ballots”, work in conjunction with Josh Benaloh, J. Alex Halderman, Ronald L. Rivest, Peter Y. A. Ryan, Philip B. Stark, Vanessa Teague, Poorvi L. Vora, and Dan S. Wallach that appeared in *Proceedings of the 2nd International Joint Conference on Electronic Voting* [35]

provide convincing evidence that announced electoral outcomes are correct.

Conversely, absent convincing evidence that the paper trail has been conserved, a manual double-check of electronic results against the paper trail will not be convincing. If the paper trail has been conserved adequately, then a full manual tally of the ballots can correct the electronic count if the electronic count is incorrect.

These considerations have led many election integrity advocates to push for a voter-verifiable paper trail (VVPAT).[2] Other techniques like *software independence* and *end-to-end verifiability* can offer far greater assurance in the accuracy of an election's outcome, but these methods have not been broadly applied.

2.1.1 Why so hard?

Several factors make it difficult to generate convincing evidence that reported results are correct. The first is the trust model.

No one is trusted In any significant election, voters, election officials, and equipment and software cannot necessarily be trusted by anyone with a stake in the outcome. Voters, operators, system designers, manufacturers, and external parties are all potential adversaries.

The need for evidence Because officials and equipment may not be trustworthy, elections should be *evidence-based*. Any observer should be able to verify the reported results based on trustworthy evidence from the voting system. Many in-person voting systems fail to provide sufficient evidence; and as we shall see Internet systems scarcely provide any at all.

The secret ballot Perhaps the most distinctive element of elections is the *secret ballot*, a critical safeguard that defends against vote selling and voter coercion. In practical terms, voters should not be able to prove how they voted to anyone, *even if they wish to do so*. This restricts the types of evidence that can be produced by the voting system. Encryption alone is not sufficient, since the voters may choose to reveal their selections in response to bribery or coercion.

The challenge of voting is thus to use fragile technology to produce trustworthy, convincing

[2] Voter-marked paper ballots or ballots marked using a ballot-marking device are preferable to VVPAT, a cash-register style printout that the voter cannot touch.

evidence of the correctness of the outcome while protecting voter *privacy* in a world *where no person or machine may be trusted*. The resulting voting system and its security features must also be *usable* by regular voters. The aim of this chapter is to explain the important requirements of secure elections.

Prior to delving into our discussion, we need to make a distinction in terminology. *Pollsite* voting systems are those in which voters record and cast ballots at predetermined locations, often in public areas with strict monitoring. *Remote* voting refers to a system where voters fill out ballots anywhere, and then send them to a central location to cast them, either physically mailing them in the case of vote-by-mail, or sending them over the Internet in the case of Internet voting.

The next section provides definitions for election integrity, including a discussion of software independence and end-to-end verifiability. Section 2.3 discusses how paper and ceremonies serve as security processes, including a discussion of the role of post-election audits. Section 2.4 discusses definitions of privacy and secrets, and Section 2.5 discusses miscellaneous definitions that help compose the overall threat surface of elections. Finally, Section 2.6 looks ahead to the rest of this dissertation and how the definitions here described factor into subsequent chapters.

2.2 Definitions of Integrity

For an election to be accepted as legitimate, the outcome should be convincing to all—and in particular to the losers—leaving no valid grounds to challenge the outcome. Whether elections are conducted by counting paper ballots by hand or using computer technology, the possibility of error or fraud necessitates assurances of the accuracy of the outcome.

It is clear that a naive introduction of computers into voting introduces the possibility of wholesale and largely undetectable fraud. If we can't detect it, how can we prevent it?

In this section, I discuss several definitions and dimensions of secure elections, ranging from broad concepts like software independence to specific ones, like collection accountability. Some of these concepts frame secure elections in different and occasionally conflicting ways, but all rely on concepts of evidence, verifiability, and integrity.

2.2.1 Software Independence

Rivest and Wack introduced a definition targeted specifically at detecting misbehavior in computer-based elections:

Definition 1. [194] A voting system is **software independent** if an undetected change or error in its software cannot cause an undetectable change or error in an election outcome.

Software independence clearly expresses that it should not be necessary to trust software to determine election outcomes, but it does not say what procedures or types of evidence should be trusted instead. A system that is not software independent *cannot* produce a convincing evidence trail, but neither can a paper-based system that does not ensure that the paper trail is complete and intact, a cryptographic voting system that relies on an invalid cryptographic assumption, or a system that relies on audit procedures but lacks a means of assuring that those procedures are properly followed.

Rivest and Wack also define a stronger form of the property that includes error recovery:

Definition 2. [194] A voting system is **strongly software independent** if it is software independent and a detected change or error in an election outcome (due to the software) can be corrected without rerunning the election.

A strongly software-independent system can recover from software errors or bugs, but that recovery in turn is generally based on some other trail of evidence.

A software independent system can be viewed as a form of tamper-evident system: a material software problem leaves a detectable trace. *Strongly* software independent systems are resilient: not only do material software problems leave a trace, the overall election system can recover from a detected problem.

One mechanism to provide software independence is to record votes on a paper record that provides physical evidence of voter's intent, can be inspected by the voter prior to casting the vote, and—if preserved intact—can later be manually audited to check the election outcome. However, as discussed in Chapter IV, a "voter-marked" and "voter-verified" paper trail are not always the same thing. Risk-limiting audits can then achieve a pre-specified level of assurance that results are correct; machine assisted risk-limiting audits [57], can help minimize the amount of labor required for legacy systems that do not provide a cast-vote record for every ballot, linked to the corresponding ballot (though caution here is also necessary, as I discuss in Chapter III).

2.2.2 End-to-end verifiability

The concern regarding fraud and desire for transparency has motivated the security and cryptography communities to develop another approach to voting system assurance: *end-to-end verifiability* (E2E-V). An election that is end-to-end verifiable achieves software independence together with the analogous notion of hardware independence as well as independence from actions of election personnel and vendors. Rather than attempting to verify thousands of lines of code or closely monitor all of the many processes in an election, E2E-V focuses on providing a means to detect errors or fraud in the process of voting and counting. The idea behind E2E-V is to enable voters themselves to monitor the integrity of the election; democracy for the people by the people, as it were. This is challenging because total transparency is not possible without undermining the secret ballot, hence the mechanisms to generate such evidence have to be carefully designed.

Definition 3. *(adapted from [31])* A voting system is **end-to-end verifiable** if it has the following three kinds of verifiability:

- **Cast as intended:** Voters can independently verify that their selections are correctly recorded.
- **Collected as cast:** Voters can independently verify that the representation of their vote is correctly collected in the tally.
- **Tallied as collected:** Anyone can verify that every well-formed, collected vote is correctly included in the tally.

If verification relies on trusting entities, software, or hardware, the voter and/or auditor should be able to choose them freely. Trusted procedures, if there are any, must be open to meaningful observation by *every* voter.

Note that the above definition allows each voter to check that her vote is correctly collected, thus ensuring that attempts to change or delete cast votes are detected. In addition, it should also be possible to check the list of voters who cast ballots, to ensure that votes are not added to the collection (i.e., to prevent ballot-box stuffing). This is called *eligibility verifiability* [133, 210].

2.2.2.1 Collection Accountability

In an E2E-V election protocol, voters can check whether their votes have been properly counted, but if they discover a problem, there may not be adequate evidence to correct it. An election system that is *collection-accountable* provides voters with evidence of any failure to collect their votes.

Definition 4. An election system is **collection accountable** if any voter who detects that her vote has not been collected has, as part of the vote-casting protocol, convincing evidence that can be presented to an independent party to demonstrate that the vote has not been collected.

Another form of evidence involves providing each voter with a code representing her votes, such that knowledge of a correct code is evidence of casting a particular vote [66]. Yet another mechanism is a suitable paper receipt. Forensic analysis may provide evidence that this receipt was not forged by a voter [26, 29].

2.2.2.2 Dispute Resolution

While accountability helps secure the election process, it is not very useful if there is no way to handle disputes. If a voter claims, on the basis of accountability checks provided by a system, that something has gone wrong, there needs to be a mechanism to address this. This is known as *dispute resolution*:

Definition 5. [126] A voting system is said to have **dispute resolution** if, when there is a dispute between two participants regarding honest participation, a third party can correctly resolve the dispute.

An alternative to dispute resolution is dispute-freeness:

Definition 6. [128] A **dispute-free** voting system has built-in prevention mechanisms that eliminate disputes among the active participants; any third party can check whether an active participant has cheated.

2.2.3 From Verifiable to Verified

Constructing a voting system that creates sufficient evidence to reveal problems is not enough on its own. That evidence must actually be used—and used appropriately—to ensure the accuracy of election outcomes.

An election result may not be verified, even if it is generated by an end-to-end verifiable voting system. For verification of the result, we need several further conditions to be satisfied:

- Enough voters and observers must be sufficiently diligent in performing the appropriate checks.
- Random audits (including those initiated by voters) must be sufficiently extensive and unpredictable that changes that affect election outcomes have a high chance of being detected.
- If checks fail, this must be reported to the authorities who, in turn, must take appropriate action.

These issues involve complex human factors, including voters' incentives to participate in verification. Existing work on this topic includes discussion of VVPAT systems [84, 104, 205] and work I present in Chapter IV, which largely confirms the fears that not enough voters participate in verifying paper trails.

A secure election system might give an individual voter assurance that her vote has not been tampered with *if* that voter performs certain checks. However, sufficiently many voters must do this in order to provide evidence that the election outcome as a whole is correct. Existing work indicates that not enough voters do this [2, 3, 5, 87, 150, 151] Combining risk-limiting audits with E2E-V systems can provide a valuable layer of protection in the case that an insufficient number of voters participate in verification, although the difficulty of reconciling E2E-V systems with real-world election systems has thus far proven too difficult to find much success, except in limited circumstances [60, 113].

Finally, another critical verification problem that has received little attention to date is how to make schemes that are recoverable in the face of errors. We do not want to have to abort and rerun an election every time a check a fails. Certain levels of detected errors can be shown to be highly unlikely if the outcome is incorrect, and hence can be tolerated. Other types and patterns of error cast doubt on the outcome and may require either full inspection or retabulation of the paper trail or, if the paper trail cannot be relied upon, a new election.

Both Küsters et al. [138] and Kiayias et al. [129] model voter-initiated auditing [28] and its implications for detection of an incorrect election result. Both definitions turn uncertainty about voter initiated auditing into a bound on the probability of detecting deviations of the announced election result from the truth.

Wallach models the effect of election official-initiated auditing both before and during an election [242], though Stark argues that such auditing will never be sufficient to provide assurance [218]. Appel, Demillo, and Stark develop notions of *contestability*, a stronger notion than software-independence that requires that a system must produce public evidence that a change in software has changed the outcome, and *defensability*, a stronger version of strong software-independence that produces a public record attesting to its correctness [15].

2.3 The Role of Paper and Ceremonies

Following security problems with direct-recording electronic voting systems (DREs) [18, 48, 155, 241], many parts of the U.S. returned to the use of paper ballots. If secure custody of the paper ballots is assumed, paper provides durable *evidence* required to determine the correctness of the election outcome. For this reason, when humans vote from untrusted computers, cryptographic voting system specifications often use paper for security, included in the notions of dispute-freeness, dispute resolution, collection accountability and accountability [137] (all as defined in Section 2.2).

Note that the standard approach to dispute resolution, based on non-repudiation, cannot be applied to the voting problem in the standard fashion, because the human voter does not have the ability to check digital signatures or digitally sign the vote (or other messages that may be part of the protocol) unassisted. Dispute-freeness or accountability are often achieved in a polling place through the use of cast paper ballots, and the evidence of their chain of custody (e.g., wet-ink signatures). However, this notion does not capture the full picture, as voters may not pay close enough attention, or if a computer system intervenes and marks their ballot incorrectly. Recent work by Appel, DeMillo, and Stark [15] argues that this prevents computer-marked ballots from providing any dispute resolution or freeness, as a voter may not be able to demonstrate the malicious change without compromising their secret vote. I discuss this further in Chapter IV

Paper provides an interface for data entry for the voter—not simply to enter the vote, but also to enter other messages that the protocol might require—and data on unforgeable paper serves many of the purposes of digitally signed data. Thus, for example, when a voter marks a Prêt à Voter [198] or Scantegrity [66] ballot, she is providing an instruction that the voting system cannot pretend was something else. The resulting vote encryption has been physically committed to by the voting system—by the mere act of printing the ballot—before the voter "casts" her vote.

Physical ceremony, such as can be witnessed while the election is ongoing, also supports verifiable cryptographic election protocols. Such ceremonies include the verification of voter credentials, any generation of randomness if required for the choice between cast and audit, any vote-encryption-verification performed by election officials, etc. Notably, these ceremonies are the type relied upon by traditional election security techniques, for instance counting ballots in public, sealing ballot containers, and so forth. The key aspect of these ceremonies is the chance for observers to see that they are properly conducted.

2.3.1 Risk-Limiting Audits

Statistical post-election audits are ceremonies that provide assurance that a reported outcome is correct, by examining some or all of an *audit trail* consisting of durable, tamper-evident, voter-verifiable records. Typically the audit trail consists of paper ballots.

The *outcome* of an election is the set of winners. An outcome is incorrect if it differs from the set of winners output by a perfectly accurate manual tabulation of the audit trail.

Definition 7. An audit of an election contest is a **risk-limiting audit (RLA)** *with risk limit* α if it has the following two properties:

1. If the reported contest outcome under audit is incorrect, the probability that the audit leads to correcting the outcome is at least $1-\alpha$.

2. The audit never indicates a need to alter a reported outcome that is correct.

(In this context, "correct" means "what a full manual tally of the paper trail would show." If the paper trail is unreliable, a RLA in general cannot detect that. RLAs should be preceded by "compliance audits" that check whether the audit trail itself is adequately reliable to determine who won.) Together, these two properties imply that post-RLA, either the reported set of winners is the set that a perfectly accurate hand count of the audit trail would show, or an event with probability no larger than α has occurred. (That event is that the outcome was incorrect, but the RLA did not lead to correcting the outcome.) RLAs amount to a limited form of probabilistic error correction: by relying on appropriate random sampling of the audit trail and hypothesis tests, they have a known minimum probability of correcting the outcome. They are not designed to ensure that the reported numerical tally is correct, only that the outcome is correct. As we shall see in Chapter VII, lowering the burden of proof on election evidence in this way is a powerful means of improving election security in a practical way.

The following procedure is a trivial RLA: with probability $1-\alpha$, perform a full manual tally of the audit trail. Amend the outcome to match the set of winners the full hand count shows if that set is different.

The art in constructing RLAs consists of maintaining the risk limit while performing *less work* than a full hand count when the outcome is correct. Typically, this involves framing the audit as a sequential test of the statistical hypothesis that the outcome is incorrect. To reject that hypothesis is to conclude that the outcome is correct. RLAs have been developed for majority contests, plurality contests, and vote-for-*k* contests and complex social choice functions including D'Hondt and other proportional representation rules—see below. RLAs have also been devised to check more than one election contest simultaneously [217]. I discuss RLAs is significantly more detail in Chapter V.

2.4 Privacy, Receipt Freeness, and Coercion Resistance

In most security applications, privacy and confidentiality are synonymous. In elections, however, privacy has numerous components that go well beyond typical confidentiality. Individual privacy can be compromised by "normal" election processes such as a unanimous result. Voters may be coerced if they can produce a proof of how they voted, even if they have to work to do so.

Privacy for votes is a means to an end: if voters don't express their true preferences then the election may not produce the right outcome. This section gives an overview of increasingly strong definitions of what it means for voters to be free of coercion.

2.4.1 Basic Confidentiality

We will take *ballot privacy* to mean that the election does not leak any information about how any voter voted beyond what can be deduced from the announced results. Confidentiality is not the only privacy requirement in elections, but even simple confidentiality poses significant challenges. It is remarkable how many deployed e-voting systems have been shown to lack even the most basic confidentiality properties (e.g., [18, 48, 62, 111, 155]).

Perhaps more discouraging to basic privacy is the fact that remote voting systems (both paper and electronic) inherently allow voters to eschew confidentiality. Because remote systems enable voters to fill out their ballots outside a controlled environment, anyone can watch over the voter's shoulder while they fill out their ballot.

In an election—unlike, say, in a financial transaction—even the candidate receiving an encrypted vote should not be able to decrypt it. Instead, an encrypted (or otherwise shrouded) vote must remain confidential to keep votes from being directly visible to election authorities.

Some systems, such as code voting [65] and the Norwegian and Swiss Internet voting schemes, defend privacy against an attacker who controls the computer used for voting; however, this relies on assumptions about the privacy and integrity of a code sheet. Some schemes, such as

JCJ/Civitas [125], obscure who has voted while providing a proof that only eligible votes were included in the tally.

Several works [85, 138], following Benaloh [33] formalize the notion of privacy as preventing an attacker from noticing when two parties swap their votes. Bernhard et al.[3] performed an analysis of game-based privacy definitions [34]

2.4.2 Everlasting Privacy

Moran and Naor expressed concern over what might happen to encrypted votes that can still be linked to their voter's name some decades into the future, and hence decrypted by superior technology (similar to forward secrecy). They define a requirement to prevent this:

Definition 8. [158] A voting scheme has **everlasting privacy** if its privacy does not depend on assumptions of cryptographic hardness.

Their solution uses perfectly hiding commitments to the votes, which are aggregated homomorphically. Instead of privacy depending upon a cryptographic hardness assumption, it is the integrity of an election that depends upon a hardness assumption; and only a real-time compromise of the assumption can have an impact.

2.4.3 Systemic Privacy Loss

We generally accept that without further information, a voter is more likely to have voted for a candidate who has received more votes, but additional data is commonly released which can further erode voter privacy. Even if we exclude privacy compromises, there are other privacy risks which must be managed. If voters achieve privacy by encrypting their selections, the holders of decryption keys can view their votes. If voters make their selections on devices out of their immediate control (e.g. official election equipment), then it is difficult to assure them that these devices are not retaining information that could later compromise their privacy. If voters make their selections on their own devices, then there is an even greater risk that these devices could be infected with malware that records (and perhaps even alters) their selections (see, for instance, the Estonian system [211]).

[3] A different Bernhard, somehow.

2.4.4 Receipt-freeness

Preventing coercion and vote-selling was considered solved with the introduction of the *Australian* ballot. The process of voting privately within a public environment where privacy can be monitored and enforced prevents improper influence. Recent systems have complicated this notion, however. If a voting protocol provides a receipt but is not carefully designed, the receipt can be a channel for information to the coercive adversary.

Benaloh and Tuinstra [32] pointed out that passive privacy is insufficient for resisting coercion in elections:

Definition 9. A voting system is **receipt free** if a voter is unable to prove how she voted *even if she actively colludes with a coercer and deviates from the protocol in order to try to produce a proof.*

Traditional elections may fail receipt-freeness too. In general, if a vote consists of a long list of choices, the number of possible votes may be much larger than the number of likely voters. This is sometimes called (a failure of) the *short ballot assumption* [197]. Prior to each election, coercers assign a particular voting pattern to each voter. When the individual votes are made public, any voter who did not cast their pattern can then be found out. This is sometimes called the *Italian attack*, after a once prevalent practice in Sicily. It can be easily mitigated when a vote can be broken up, but is difficult to mitigate in systems like IRV in which the vote is complex but must be kept together. Mitigations are discussed later on in Chapter V.

Incoercibility has been defined and examined in the universally composable framework in the context of general multiparty computation [59, 228]. These definitions sidestep the question of whether the voting function itself allows coercion (by publishing individual complex ballots, or by revealing a unanimous result for example)—they examine whether the protocol introduces additional opportunities for coercion. With some exceptions (such as [11]), they usually focus on a passive notion of receipt-freeness, which is not strong enough for voting.

2.4.5 Coercion Resistance

Schemes can be receipt-free, but not entirely resistant to coercion. Schemes like Prêt à Voter [198] that rely on randomization for receipt-freeness can be susceptible to *forced randomization*, where a coercer forces a voter to always choose the first choice on the ballot. Due to randomized candidate order, the resulting vote will be randomly distributed. If a specific group of voters are coerced in this way, it can have a disproportionate impact on the election outcome. If voting rolls are public and voting is not mandatory, this has an effect equivalent to *forced abstention*, wherein a coercer refuses to let a voter vote. Schemes that rely on credentialing are also susceptible to coercion by *forced surrender of credentials*.

One way to fully resist forced abstention is to obscure who voted. However, this is difficult to reconcile with the opportunity to verify that only eligible voters have voted (eligibility verifiability), though some schemes achieve both [109].

Moran and Naor [158] provide a strong definition of receipt freeness in which a voter may deviate actively from the protocol in order to convince a coercer that she obeyed. Their model accommodates forced randomization. A scheme is resistant to coercion if the voter can always pretend to have obeyed while actually voting as she likes.

Definition 10. A voting scheme is **coercion resistant** if there exists a way for a coerced voter to cast her vote such that her coercer cannot distinguish whether or not she followed the coercer's instructions.

Coercion resistance is defined in [125] to include receipt freeness and defence against forced-randomization, forced abstention and the forced surrender of credentials. More general definitions include [139], which incorporates all these attacks along with Moran and Naor's notion of a coercion resistance strategy.

Note that if the coercer can monitor the voter throughout the vote casting period, then resistance is futile. For in-person voting, we assume that the voter is isolated from any coercer while she is in the booth (although this is questionable in the era of mobile phones). For remote voting, we need to assume that voters will have some time when they can interact with the voting system (or the credential-granting system) unobserved.

2.4.5.1 More Coercion Considerations

Some authors have tried to provide some protection against coercion without achieving full coercion resistance. *Caveat coercitor* [107] proposes the notion of *coercion evidence* and allows voters to cast multiple votes using the same credential. I develop a definition of *coercion-hardness* in Chapter VII, where voters may be coerced, but not so much that it can overturn an election outcome.

2.5 Other Security Considerations

Now that I have discussed what it means for elections to have integrity, verifiability, and privacy, here I briefly cover some other facets that are just as important to carrying out a secure election. Many of the topics discussed here are relatively understudied, and as we shall see in subsequent chapters, these blind spots can have a dramatic impact on whether an election is secure in the real world.

2.5.1 Voter Authentication

A significant challenge for election systems is the credentialing of voters to ensure that all eligible voters, and no one else, can cast votes. This presents numerous questions: what kinds of credentials should be used? How should they be issued? Can they be revoked or de-activated? Are credentials good for a single election or for an extended period? How difficult are they to share, transfer, steal, or forge? Can the ability to create genuine-looking forgeries help prevent coercion? These questions must be answered carefully, and until they are satisfied for remote voting, pollsite voting is the only robust way to address these questions—and even then, in-person credentialing is subject to forgery, distribution, and revocation concerns (for instance, the Dominican Republic recently held a pollsite election where voters openly sold their credentials [98]). In the U.S., there is concern that requiring in-person credentialing, in the form of voter ID, disenfranchises legitimate voters, and in recent years in vote-by-mail as well (which I discuss in Chapter VII).

2.5.2 Availability

Denial-of-Service (DoS) is an ever-present threat to elections which can be mitigated but never fully eliminated. A simple service outage can disenfranchise voters, and the threat of attack from foreign state-level adversaries is a pressing concern. Indeed, one of the countries that regularly uses Internet voting, Estonia, has been subject to malicious outages [225].

A variant of DoS specific to the context of elections is *selective DoS*, which presents a fundamentally different threat than general DoS. Voting populations are rarely homogeneous, and disruption of service, for instance, in urban (or rural) areas can skew results and potentially change election outcomes. If DoS cannot be entirely eliminated, can service standards be prescribed so that if an outcome falls below the standards it is vacated? Should these standards be dependent on the reported margin of victory? What, if any, recovery methods are possible? Because elections are more vulnerable to minor perturbations than most other settings, selective DoS is a concern which cannot be ignored.

2.5.3 Usability

A voting system must be *usable* by voters, poll-workers, election officials, observers, and so on. Voters who may not be computer literate—and sometimes not literate at all—should be able to vote with very low error rates. Although some error is regarded as inevitable, it is also critical that the interface not drive errors in a particular direction. For instance, a list of candidates that crosses a page boundary could cause the candidates on the second page to be missed [24]. Whatever security mechanisms we add to the voting process should operate without degrading usability, otherwise the resulting system will likely be unacceptable. A full treatment of usability in voting is beyond the scope of this chapter. However, we note that E2E-V systems (and I-voting systems, even when not E2E-V) add additional processes for voters and poll workers to follow. If verification processes can't be used properly by real voters, the outcome will not be properly verified. One great advantage of statistical audits is to shift complexity from voters to auditors.

An historical example involves direct recording electronic (DRE) voting systems, which were widely adopted by many voting jurisdictions in the United States after the year 2000 (and also in places like India [247]). These systems are essentially computers that simulate a paper ballot experience, allowing voters to make selections on a digital ballot. Everett et al. [96] showed that "summary screens" with deliberately introduced errors are not noticed by greater than half of their test subjects. If users' attention to this detail were required in order to maintain system security, then DRE summary screens cannot suffice. I discuss this further in Chapter IV.

2.5.4 Local Regulatory Requirements

A variety of other mechanical requirements are often imposed by legal requirements that vary among jurisdictions. For example:

- Allowing voters to "write-in" vote for a candidate not listed on the ballot.
- Mandating the use of paper ballots (in some states without unique identifying marks or serial numbers; in other states *requiring* such marks)
- Mandating the use of certain social choice functions (see 2.5.5 below).
- Supporting absentee voting.
- Requiring or forbidding "ballot rotation" (listing the candidates in different orders in different jurisdictions).
- Requiring that voting equipment be certified under government guidelines.

2.5.5 Complex Election Methods

Many countries allow voters to *select, score, or rank* candidates or parties. Votes can then be tallied in a variety of complex ways [50, 200]. None of the requirements for privacy, coercion-resistance, or the provision of verifiable evidence change. However, many tools that achieve these properties for traditional "first-past-the-post" elections need to be redesigned.

An election method might be complex at the voting or the tallying end. For example, party-list methods such as D'Hondt and Sainte-Laguë have simple voting, in which voters select their candidate or party, but complex proportional seat allocation. Borda, Range Voting, and Approval

Voting allow votes to be quite expressive but are simple to tally by addition. Condorcet's method and related functions [204, 224] can be arbitrarily complex, as they can combine with any social choice function. Instant Runoff Voting (IRV) and the Single Transferable Vote (STV) are both expressive and complicated to tally.

2.6 A Look Ahead

Now that we have examined some basic definitions of election security and the factors that impact them, we can turn our discussion to practical examples. The key definitions that we will examine in the rest of this work include software independence, verifiability, and coercion resistance.

CHAPTER III

Why Paper Matters[1]

3.1 Introduction

As we have just discussed, elections that cannot provide sufficient evidence of their results may fail to adequately gain public confidence in their outcomes. Numerous solutions have been posited to this problem [35], but none has been as elegant, efficient, and immediately practical as post-election audits [112, 143, 215]. These audits—in particular, ones that seek to limit the risk of confirming an outcome that resulted from undue manipulation—are one of the most important layers of defense for election security [160].

Risk-limiting audits (RLAs) rely on sampling robust, independent evidence trails created by voter-verified paper ballots. However, the practical constraints often faced in election jurisdictions, coupled with the fact that the statistical knowledge required to implement RLAs correctly, make implementing RLAs in a meaningful way difficult. Indeed, this has been borne out by the fact that RLAs, though nearly fifteen years old at the time of writing, are only just now starting to gain adoption in the U.S. and elsewhere.

However, other types of post-election audits have gained popularity in the marketplace in their stead. In particular, Clear Ballot, an election technology vendor, pioneered audit software designed to perform audits of *images* of ballots which have been scanned and tabulated, which we shall refer

[1] This chapter is based on “UnclearBallot: Automated Ballot Image Manipulation”, work in conjunction with Kartikeya Kandula, Jeremy Wink, and J. Alex Halderman that appeared in *International Joint Conference on Electronic Voting* [36]

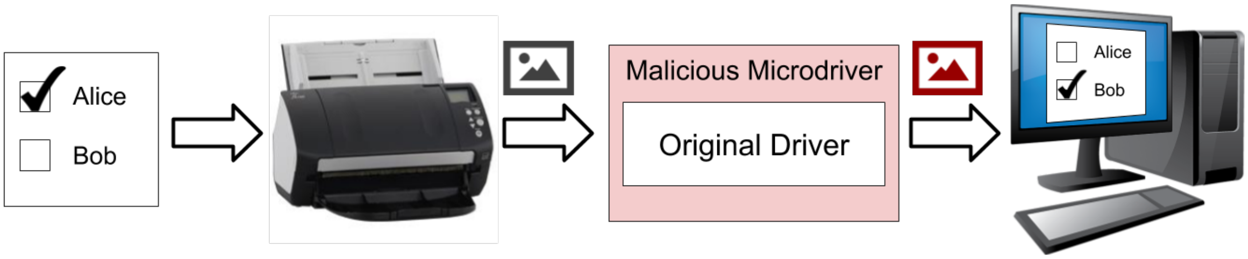


Figure 3.1: **Attack overview** — A voter's paper ballot is scanned by a ballot tabulator, producing a digital image. Malware in the tabulator—in our proof-of-concept, a microdriver that wraps the scanner device driver—alters the ballot image before it is counted or stored. A digital audit shows only the manipulated image.

to as "image audits". Other vendors have adopted support for this kind of audit, and one U.S. state, Maryland, relies on image audits to provide assurances of its election results [162].

While image audits can help detect human error and aid in adjudicating mismarked ballots, we show that they cannot provide the same level of security assurance as audits of physical ballots. Since ballot images are disconnected from the actual source of truth—physical paper ballots—they do not necessarily provide reliable evidence of the outcome of an election under adversarial conditions. To put it another way, as image audits fundamentally rely on software, they cannot provide the software independence properties that audits of physical paper can.

In this chapter, we present UnclearBallot, an attack that defeats image audits by automatically manipulating ballot images as they are scanned. This attack leverages the same computer vision approaches used by ballot scanners to detect voter selections, but adds the ability to move marks from one target area to another. The method is robust to inconsistent or invalid marks, and can be adapted to many ballot styles.

We validate this attack against a corpus of over 180,000 ballot images from the 2018 election in Clackamas County, Oregon, and find that UnclearBallot can move marks on 34% of the ballots while leaving no visible anomalies. We also test the attack's flexibility using six widely used styles of paper ballots, and its robustness to invalid votes using an established taxonomy of voter marks. As a proof-of-concept, we implement the attack in the form of a malicious Windows scanner driver, which was tested using a commercial-off-the-shelf scanner certified for use in elections by the U.S.

Election Assistance Commission.

UnclearBallot illustrates that post-election audits in traditional voting systems must involve rigorous examination of *physical ballots*, rather than ballot images, if they are to provide a strong security guarantee. Without an examination of the physical evidence, it will be difficult if not impossible to assure that computer-based tampering has not occurred. Essentially this is a case-study in how known election security techniques can be misapplied or misunderstood to provide properties that they do not.

The remainder of this chapter is organized as follows: Section 3.2 provides background on image audits, ballot scanners, and image processing techniques used to implement the attack. Section 3.3 describes the attack scenarios against optical scanners and image audits. Section 3.4 explains the methodology of the attack. In Section 3.5 we present data indicating that the attack can be robust to various ballot styles and voter marks. Section 3.6 contextualizes our attack and discusses mitigations. We conclude in Section 3.7.

3.2 Background

The attack takes advantage of two aspects of optical scanner image audits: the scanning and image processing techniques used by scanners, and the reliance on scanned images by image audits as the only source of truth. Here we provide a brief discussion of both.

3.2.1 Ballot Images

Jones [124] put forth an analysis of the way that ballot scanners work, particularly the mark-sense variety that is most common today. All optical scanners currently sold to jurisdictions, as well as the vast majority of scanners used in practice in the U.S., rely on mark-sense technology [234]. Scanners first create a high-resolution image of a ballot as it is fed past a scan head. Software then analyzes the image to identify dark areas where marks have been made by the voter.[2] Once marks have been detected, systems may use template matching to translate marks into votes for specific candidates, typically relying on a barcode or other identifier on the ballot that specifies a ballot style

[2] The details of how marks are identified vary by hardware and scanning algorithm. See [69] for an example.

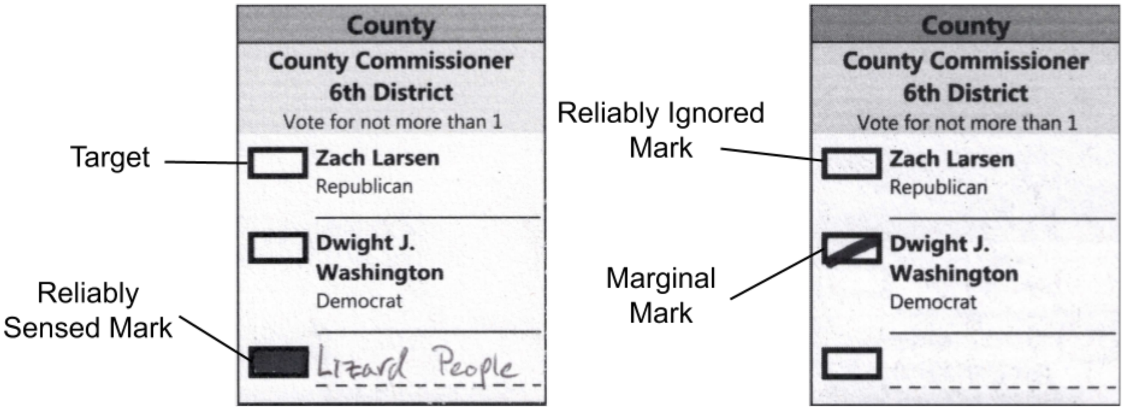


Figure 3.2: **Terms for parts of a marked ballot**, following Jones [124].

to match to the scanned image.

Detecting and interpreting voter marks can be a difficult process, as voters exhibit a wide range of marking and non-marking behavior, including not filling in targets all the way, resting their pens inside targets, or marking outside the target. The terms Jones developed to refer to the ballot and marks are illustrated in Figure 3.2. Marks that adequately fill the target and are unambiguously interpreted as votes by the scanner are called *reliably sensed* marks, and targets that are unambiguously not filled and therefore not counted are *reliably ignored* marks. Marks of other types are deemed *marginal*, as a scanner may read or ignore them. Moreover, whether a mark should be counted as a vote is frequently governed by local election statute, so some marginal marks may be unambiguously counted or ignored under the law, even if not by the scanner.

Bajcsy et al. [20] further develops a systematization of marginal marks and develops some improvements on mark-detection algorithms to better account for them. An illustration of Bajcsy et al.'s taxonomy is shown in Figure 3.3. Ji et al. [121] discuss different types of voter marks as applied to write-in votes, as well as developing an automated process for detecting and tabulating write-in selections.

3.2.2 Image Audits

Risk-limiting post-election audits rely on physical examination of a statistical sample of voter-marked ballots [140, 142, 215, 217]. However, this can create logistical challenges for election

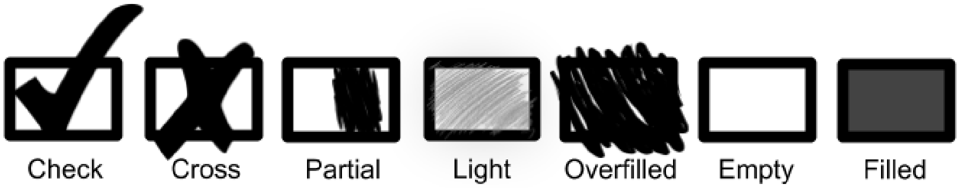


Figure 3.3: **Taxonomy of voter marks** adapted from Bajcsy [20], including the five leftmost marks that may be considered marginal marks.

officials, which has prompted some to propose relaxations to traditional audit requirements. To reduce workload, canvass audits and recounts in many states rely on retabulation of ballots through optical scanners (see the 2016 Wisconsin recount, for example [156]).

Some election vendors take retabulation audits a step further: rather than physically rescan the ballots, the voting system makes available images of all the ballots for independent evaluation after the election [73, 89, 226].[3] While the exact properties of these kinds of image audits vary by vendor, they typically rely on automatically retabulating all or some images of cast ballots, as well as electronic adjudication for ballots with marginal marks. These “audits” never examine the physical paper trail of ballots, which our attack exploits.

Several jurisdictions have relied on these image audits, including Cambridge, Ontario, which used Dominion’s AuditMark [90], and the U.S. state of Maryland, which uses Clear Ballot’s ClearAudit [153]. Maryland has also codified image audits into its election code, requiring that an image audit be performed after every election [152].

3.3 Attack Scenarios

Elections in which voters make their selections on a physical ballot are frequently held as the gold standard for conducting a secure election [160]. However, the property that contributes most to their security, software independence [194], only exists if records computed by software are checked against records that cannot be altered by software without detection. Image audits enable election officials to view images of ballots and compare them with the election systems’ representation of the particular ballot they are viewing (called a cast vote record or CVR). While these two trails

[3] While the review is made available to the public, the actual images themselves are seldom published in full out of concern for voter anonymity.

of evidence may be independent from each other (for example, Clear Ballot's ClearAudit [73] technology can be used to audit a tabulation performed by a different election system altogether), they are not software independent. A clever attacker can exploit the reliance on software by both evidence trails to defeat detection.

To successfully defend against an attack, an image audit must assume one of two things. If ballot images are not considered the only source of truth for the election, and image audit must assume an attacker cannot modify both the CVR (or other source of truth) and the ballot images in a coordinated way. If, on the other hand, the ballot image is the sole source of truth, the image audit must assume that ballot images cannot be altered in an undetectable way.

In either case, there are two outcomes an attacker can seek to achieve: changing the election outcome to their preference, or disrupting the election from passing the audit without changing its outcome. The latter requires the attacker to modify the election evidence trail in some way that is detected by the audit, and such modification is a subset of the attack required to achieve the former. As such, we will only focus on changing the election outcome.

An attacker can change the election outcome either with or without detection. In the case where ballot images are the source of truth for an election, simply modifying the images to the attackers' preferred outcome will be detected by the image audit and overturn the election result provided by the tabulation software. The actions to achieve this are a subset of the actions required to change the election outcome without detection, so we will only discuss ways to change the election outcome without detection.

To surreptitiously change the outcome of the election in the presence of an image audit, the attacker must alter both the tabulation result as well as the ballot images themselves. Researchers have documented numerous vulnerabilities that would allow an attacker to infect voting equipment and change tabulation results (see [18, 48, 155] among others), so we focus on the feasibility of manipulating ballot images once an attacker has successfully infected a machine where they are stored or processed.

The most straightforward attack scenario occurs when the ballot images are created by the same equipment that produces the CVR. In this case, the attacker can simply infect the scanner or tabulator with malware that corrupts both the CVR and the images at the same time. The attack could change the image before the tabulator processes it to generate the CVR, or directly alter both sets of records.

In some jurisdictions, the ballot images that are audited are collected in a separate process from tabulation—that is, by scanning the ballots again, as in Maryland's use of ClearAudit from 2016 [153]. In this case, the adversary has to separately attack both processes, and has to coordinate the cheating to avoid mismatches between the initial tally and the altered ballot images.

Depending on the timing of the audit, manipulation of ballot images need not be done on the fly. For example, if the ballot images are created during tabulation but the image audit does not occur until well after the election, an attacker could modify the ballot images while they are in storage.

For ease of explication, the discussion that follows assumes that ballot images are created at the time of tabulation, in a single scan. The attack we develop targets a tabulation machine and manipulates each ballot online as it is scanned.

3.4 Methodology

To automatically modify ballot images, an attacker can take a few approaches. One approach would be to completely replace the ballot images with ballots filled in by the attacker. However, this risks being detected if many ballots have the same handwriting, and requires sneaking these relatively large data files into the election system without being detected. For these reasons, we investigate an alternative approach: automatically and selectively doctoring the ballot scans to change the vote selections they depict.

For the attack to work successfully, we need to move voter marks to other targets without creating visible artifacts or inconsistencies. We must be able to dynamically detect target areas and marks, alter marks in a way that is consistent with the voter's other marks, and do so in a way that is undetectable to the human eye. However, there is a key insight that works in the adversary's favor: an attacker seeking to alter election results does not have to be able to change *all* ballots undetectably, only sufficiently many to swing the result. This means that the attacker's manipulation strategy does not need to be able to change *every* mark—it merely has to reliably detect *which* marks it can safely alter and change enough of them to decide the election result.

3.4.1 Reading the ballot

To interpret ballot information, we rely on the same techniques that ballot scanners use to convert paper ballots into digital representations. Attackers have access to the ballot templates, as jurisdictions publish sample ballots well ahead of scheduled elections. Using template matching, an attacker does not have to perform any kind of sophisticated character recognition, they simply have to find target areas and then detect which of the targets are filled.

Our procedure to read a ballot is illustrated in Figure 3.4. First, we perform template matching to extract each individual race within a ballot. Next, we use OpenCV's [49] implementation of the Hough transform to detect straight lines that separate candidates and break the race into individual panes for each candidate. Notably, the first candidate in each race may have the race title and extra information in it (see Figure 3.4c), which is cropped out based on white space.

Target areas are typically printed on the ballot as either ovals or rectangles. To detect them, we construct a bounding box around the target by scanning horizontally from the left of the race and then vertically from the bottom up, and compute pixel density values. The bounds are set to the coordinates where the density values first increase and last decrease. Once we have detected all the target areas, we compute the average pixel density of the area within the bounding box to determine whether or not a target area is marked. We then use our template to convert marks into votes for candidates.

3.4.2 Changing marks

Once we have identified which candidate was marked by the voter, we can move the mark to one of the other target locations we identified. If the vote is for a candidate the attacker would like to receive fewer votes—or if it is not a vote for a candidate they would like to win—the attacker can simply swap the pixels within the bounding boxes of the voter's marked candidate and an unmarked

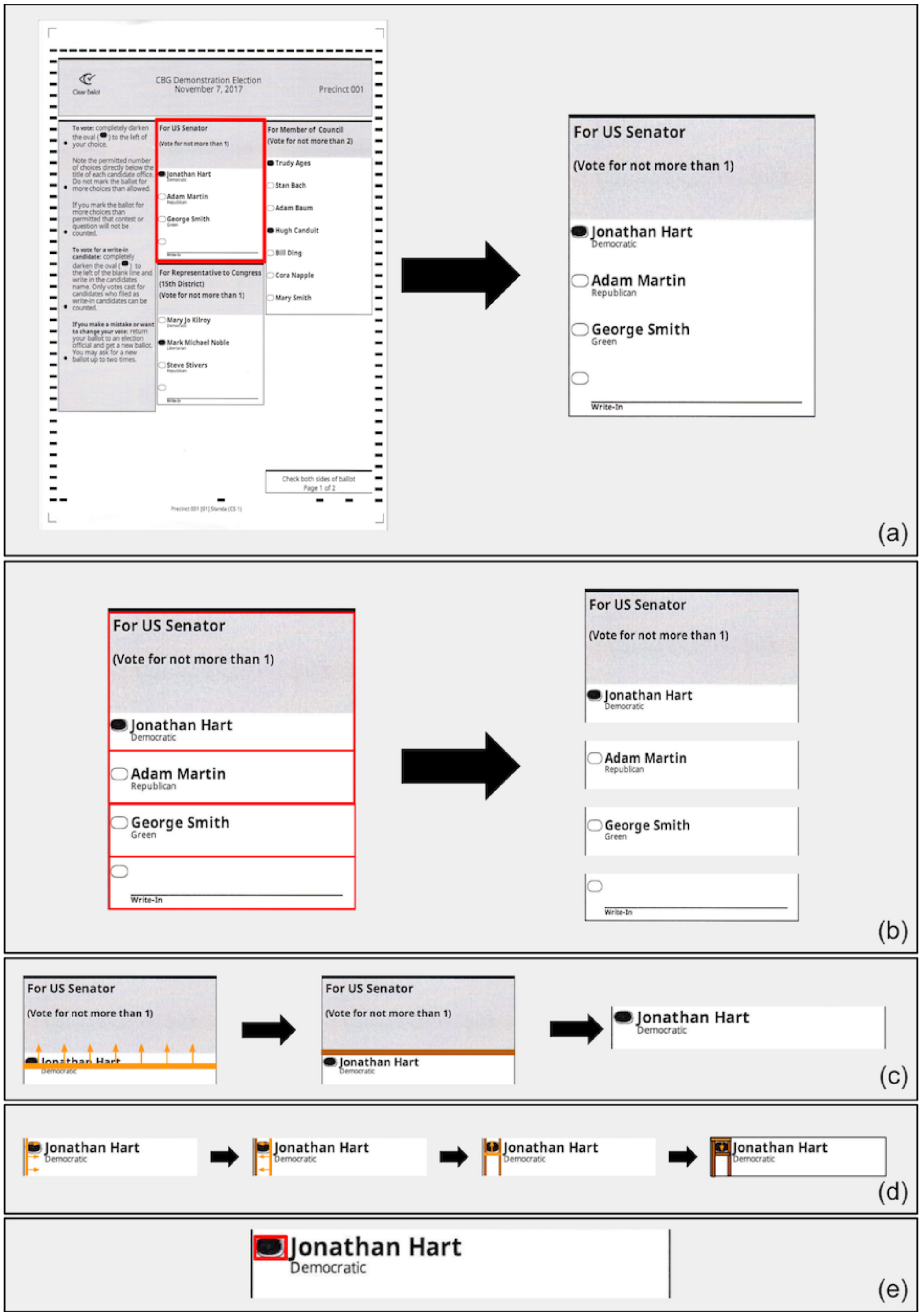


Figure 3.4: **Ballot manipulation algorithm** — First, (a) we apply template matching to extract the race we intend to alter. Then, (b) we use Hough line transforms to separate each candidate. If the first candidate has a race title box, (c) we remove it by computing the pixel intensity differences across a straight line swept vertically from the bottom. For each candidate, (d) we identify the target and mark (if present) by doing four linear sweeps and taking pixel intensity. Finally, (e) we identify and move the mark. At each step we apply tests to detect and skip ballots where the algorithm might leave artifacts.

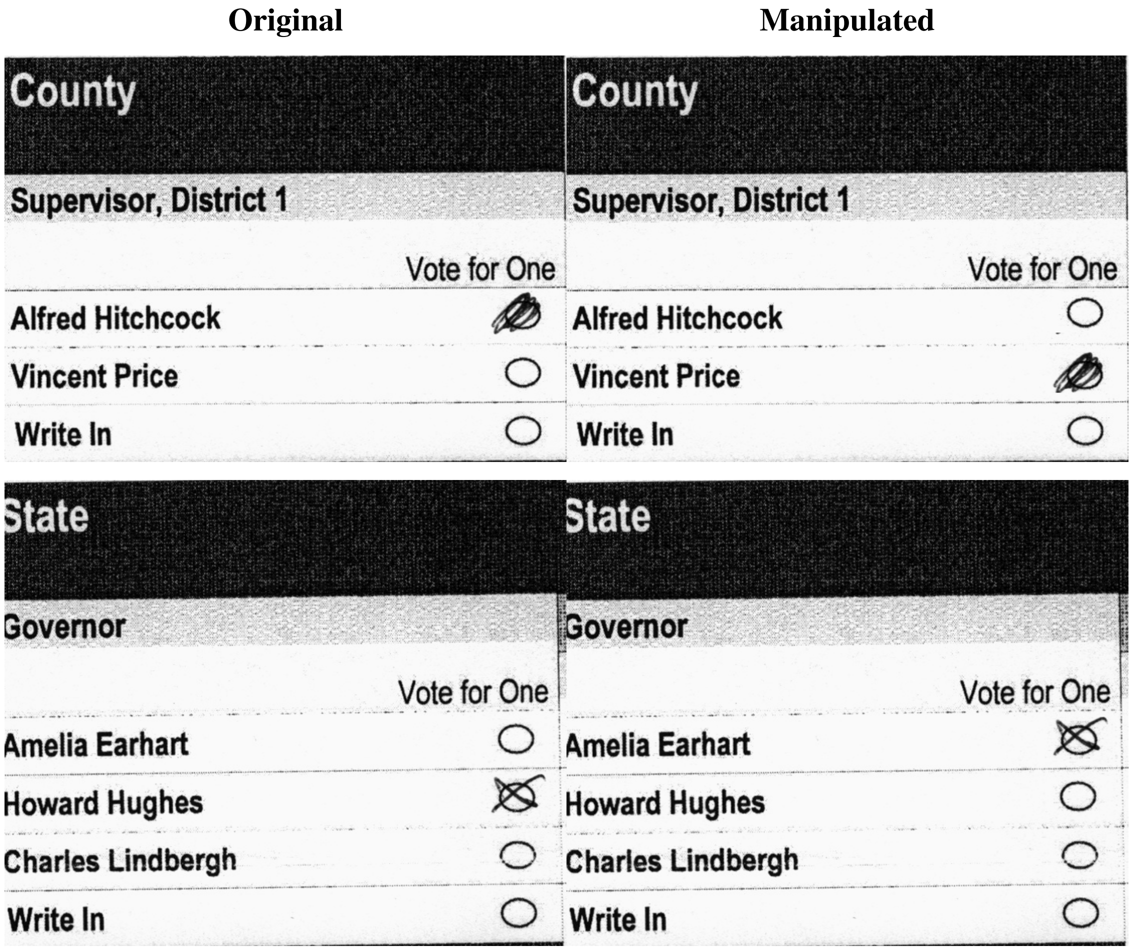


Figure 3.5: **Automatically moving voter marks** — UnclearBallot seamlessly moves marks to the attacker's preferred candidate while preserving the voter's marking style. It is effective for a wide variety of marks and ballot designs. In the examples above, original ballot scans are shown on the left and manipulated images on the right.

candidate. By moving marks on each ballot separately, we ensure that the voter's particular style of filling in an oval is preserved and consistent across the ballot. Figure 3.5 shows some marks swapped by our algorithm, and how the voters original mark is completely preserved in the process.

3.4.3 UnclearBallot

To illustrate the attack, we created UnclearBallot, a proof-of-concept implementation packaged as a malicious Windows scanner driver, which consists of 398 lines of C++ and Python. We tested it with a Fujistu fi-7180 scanner (shown in Figure 3.6), which is federally certified for use in U.S. elections as part of Clear Ballot's ClearVote system [231]. These scanners are typically used to

Figure 3.6: The **Fujitsu fi-7180 scanner** we used to test our attack has been certified by the U.S. Election Assistance Commission for use in voting systems. Our proof-of-concept implementation is a malicious scanner driver that alters ballots on the fly.

handle small volumes of absentee ballots, and must be attached to a Windows workstation that runs the tabulation software.

The UnclearBallot driver wraps the stock scanner driver and alters images from the scanner before they reach the election management application. We chose this approach for simplicity, as the Windows driver stack is relatively easy to work with, but the attack could also be implemented at other layers of the computing stack. For instance, it could be even harder to detect if implemented as a malicious change to the scanner's embedded firmware. Alternatively, it could could be engineered as a modification to the tabulation software itself.

Once a ballot is scanned, the resulting bitmap is sent to our image processing software, which manipulates the ballot in the way described in Section 3.4.1. Prior to the election, the attacker specifies the ballot template, which race they would like to affect, and by how much. While ballots are being scanned, the software keeps a running tally of the actual ballot results, and changes ballot images on the fly to achieve the desired election outcome. To avoid detection, attackers can specify just enough manipulated images so that the race outcome is changed.

3.5 Evaluation

We evaluated the performance and effectiveness of UnclearBallot using two sets of experiments. In the first set of experiments, we marked different ballot styles by hand using types of marks taxonomized by Bajcsy et al. [20]. In the second set of experiments, we processed 181,541 ballots from the 2018 election in Clackamas County, Oregon.

3.5.1 Testing Across Ballot Styles

In order for our application to succeed at its goal (surreptitiously changing enough scanned ballots to achieve a chosen election outcome), it must be able to detect marks that constitute valid votes as well as distinguish marks which would be noticeable if moved. The marks in the latter case represent a larger set than just marginal marks, as they may indeed be completely valid votes, but considered invalid by our mark-moving algorithm. For example, if we were to swap the targets on a ballot where the user put a check through their target, we may leave a significant percentage of the check around the original target when swapping. The same applies for marked ballots where the filled in area extends into the candidate's name, which could lead our algorithm to swap over parts of the candidate's name when manipulating the image.

To detect anomalies for invalid ballots, we leverage the same intensity checking algorithm that first found the marked areas. The program checks if the width or height is abnormally large, which would indicate an overfilled target, as well as if there are too few or too many areas of high intensity, which would indicate no target or too many targets are filled out. If the program detects an invalid ballot, it will not be modified by the program.

To show our attack is replicable on a variety of different ballot styles, we modified our program to work on six different sample ballot styles, shown in Figure 3.7. The ballots we tested come from the four largest election vendors in the U.S. (ES&S, Hart InterCivic, Dominion, and Clear Ballot), as well as two older styles of ballots from Hart and Diebold.

Our first experiment was designed to characterize the technique's effectiveness across a range of ballot styles and with both regular and marginal marks. We prepared 720 marked contests, split evenly among the six ballot styles shown in Figure 3.7. For each style, we marked 60 contests with what Bajcsy [20] calls "Filled" marks, i.e. reliably detected marks that should be moved by our attack. We marked another 60 ballots in each ballot style with marginal marks, ten each for the five kinds of marginal marks shown in Figure 3.2 and ten empty marks.

Because the runtime of the template matching step of our algorithm is highly dependent on

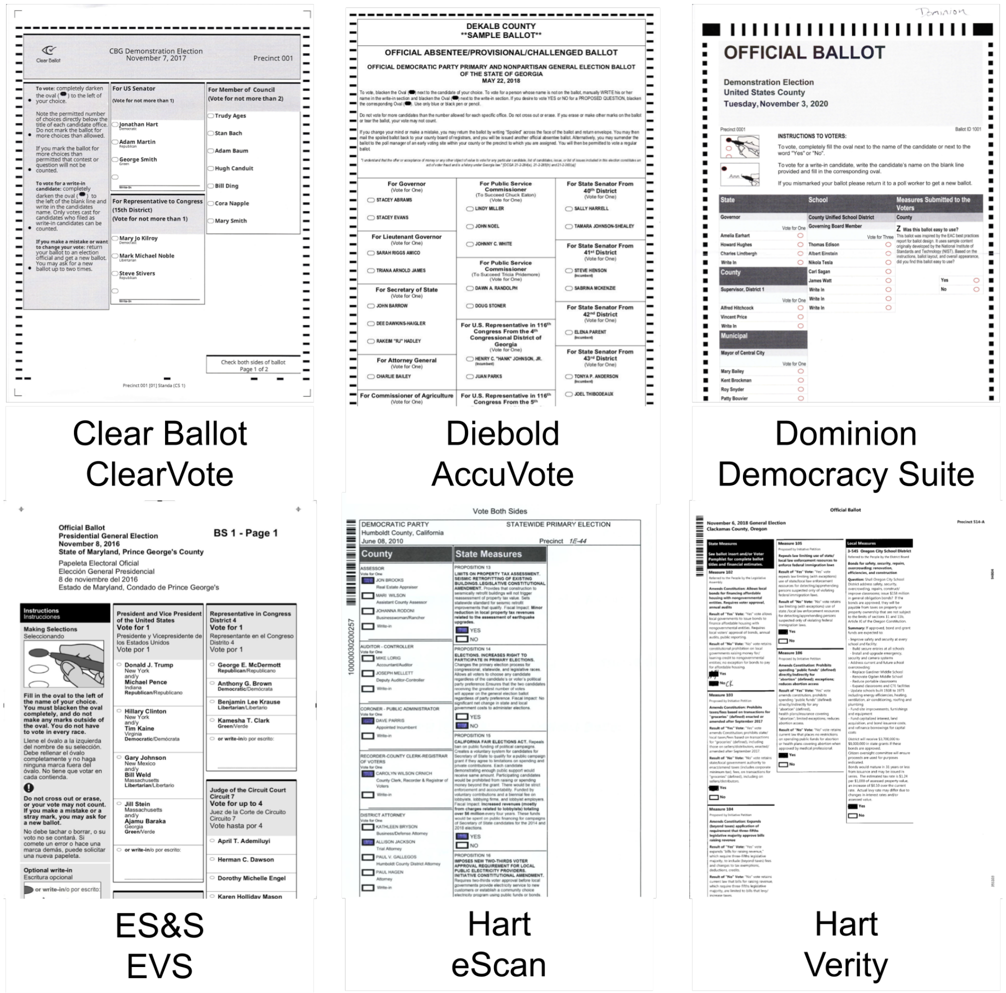


Figure 3.7: **Ballots Styles** — We tested ballot designs from five U.S. voting system vendors: Clear Ballot, Diebold, Dominion, ES&S, and Hart (two styles, eScan and Verity).

customization for the particular races on a ballot, we opted to skip it for this experiment. Rather than marking full ballots, we marked cropped races from each ballot style and then ran them through our program. We then manually checked to ensure that the races the program moved were not detectable by inspection. Results for these experiments are shown in Table 3.1.

Despite rejecting some valid ballots, our program is still able to confidently swap a majority of valid votes. In a real attack, only a small percentage of votes would need to actually be modified, a task easily accomplished by our program. Our program also correctly catches all votes that we have deemed invalid for swapping. This would make it unlikely to be detected in an image audit.

Dominion ballots saw a much higher rate of invalid mark moving, and Diebold and Dominion

Ballot Style	Invalid Marks			Valid Marks			Time/Success
	Skipped	Success	Failure	Skipped	Success	Failure	
Clear Ballot	55	5	0	26	34	0	25 ms
Diebold	60	0	0	6	54	0	11 ms
Dominion	38	22	0	7	53	0	30 ms
ES&S	52	8	0	29	31	0	54 ms
Hart (eScan)	60	0	0	38	22	0	46 ms
Hart (Verity)	60	0	0	27	33	0	21 ms

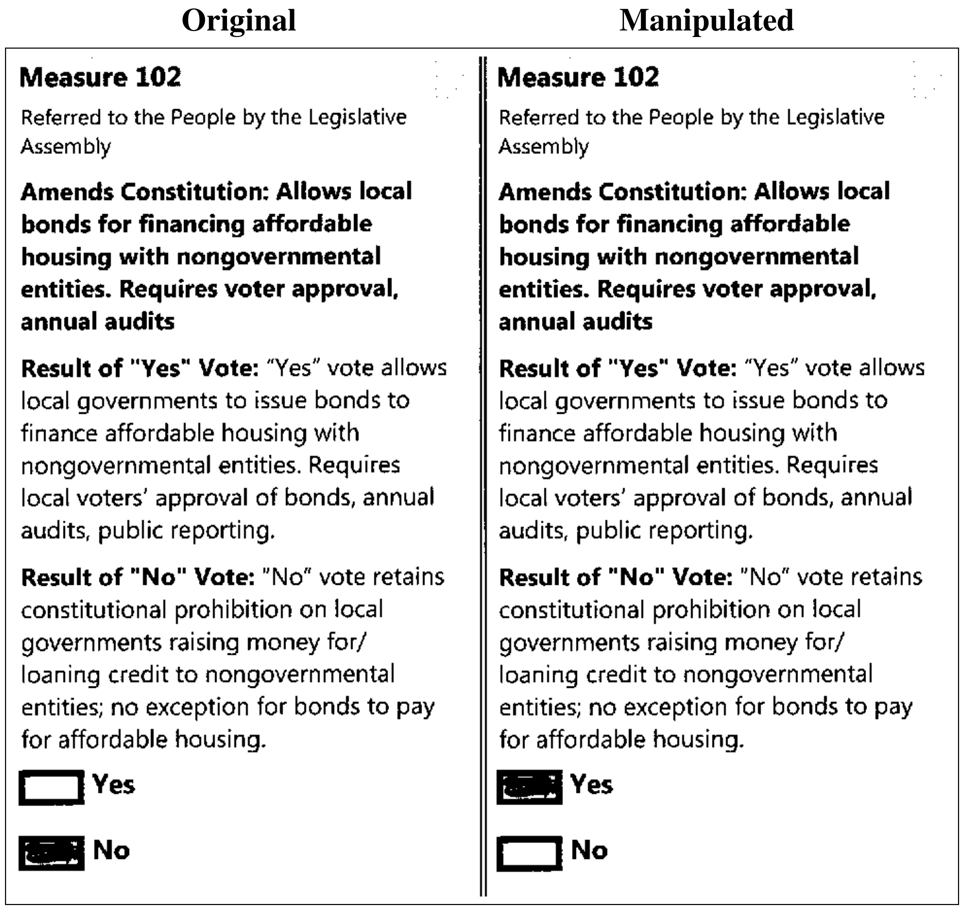
Table 3.1: **Performance of UnclearBallot** — We tested how accurately our software could manipulate voter marks for a variety of ballot styles using equal numbers of invalid and valid marks. The table shows how often the system skipped a mark, successfully altered one, or erroneously created artifacts we deemed to be visible upon manual inspection. We also report the mean processing time for successfully manipulated races, excluding template matching.

ballots saw a much higher rate of valid mark moving. This is likely due to the placement of targets: on the Dominion ballots, the mark is right justified, separating it significantly from candidate label information, as can be seen in Figure 3.7. Similarly, the Diebold ballot provides more space around the target and less candidate information that can be intercepted by marks, which would cause Unclear Ballot to skip moving the mark.

In an online attack scenario (such as if a human is waiting to see the output from the scanner), the attacker needs to be able to modify ballot scans quickly enough not to be noticed. Factors which might affect how quickly our program can process and manipulate ballots include ballot style, layout, and type of mark. During the accuracy experiment just described, we collected timing data for successfully manipulated ballot, and report the results in Table 3.1. The results show that after the target race has been extracted, the algorithm completes extremely quickly for all tested ballot styles. We present additional timing data at the end of the following section.

3.5.2 Testing with Real Voted Ballots

To assess the effectiveness of UnclearBallot in a real election, we used a corpus of scans of 181,541 real ballots from the November 6, 2018, General Election in Clackamas County, Oregon, which were made available by Election Integrity Oregon [93]. Like all of Oregon, Clackamas



Original

Measure 102

Referred to the People by the Legislative Assembly

Amends Constitution: Allows local bonds for financing affordable housing with nongovernmental entities. Requires voter approval, annual audits

Result of "Yes" Vote: "Yes" vote allows local governments to issue bonds to finance affordable housing with nongovernmental entities. Requires local voters' approval of bonds, annual audits, public reporting.

Result of "No" Vote: "No" vote retains constitutional prohibition on local governments raising money for/ loaning credit to nongovernmental entities; no exception for bonds to pay for affordable housing.

☐ **Yes**

■ **No**

Manipulated

Measure 102

Referred to the People by the Legislative Assembly

Amends Constitution: Allows local bonds for financing affordable housing with nongovernmental entities. Requires voter approval, annual audits

Result of "Yes" Vote: "Yes" vote allows local governments to issue bonds to finance affordable housing with nongovernmental entities. Requires local voters' approval of bonds, annual audits, public reporting.

Result of "No" Vote: "No" vote retains constitutional prohibition on local governments raising money for/ loaning credit to nongovernmental entities; no exception for bonds to pay for affordable housing.

■ **Yes**

☐ **No**

Figure 3.8: **Attacking Real Ballots** — Using 181,541 images of voted ballots from Clackamas County, Oregon, we attempted to change voters' selections for the ballot measure shown above. UnclearBallot determined that it could safely alter 34% of the ballots. For reference, Measure 102 passed by a margin of 5%, well within range of manipulation [70]. We inspected 1,000 of them to verify that the manipulation left no obvious artifacts.

County uses vote-by-mail as its primary voting method, and votes are centrally counted using optical scanners. All images were Hart Verity-style ballots, as shown in Figure 3.7.

We selected a ballot measure that appeared on all the ballots (Figure 3.8) and attempted to change each voter's selection. UnclearBallot rejected 20,117 (11%) of the ballots because it could not locate the target contest. We examined a subset of the rejected ballots and found that they contained glitches introduced during scanning (such as vertical lines running the length of the ballot), which interfered with the Hough transform.

To simulate a real attacker, we configured UnclearBallot with conservative parameters, so that it would only modify marks when there was high confidence that the alteration would not be noticeable. As a result, it would only manipulate marks that were nearly perfectly filled in. In most cases, marks that were skipped extended well beyond the target, but the program also skipped undervotes, overvotes, or mislabeled scans. Under these parameters, the program altered the target contest in 62,400 (34%) of the ballot images.

Two authors independently inspected a random sample of 1,000 altered ballots to check whether any contained artifacts that would be noticeable to an attentive observer. Such artifacts might include marks which were unnaturally cut off, visible discontinuities in pixel darkness (i.e. dark lines around moved marks), and so on. If these artifacts were seen during an audit, officials might recheck all of the physical ballots and reverse the effects of the attack. None of the altered ballots we inspected contained noticeable evidence of manipulation.

We also collected timing data while processing Clackamas County ballots. Running on a system with a 4-core Intel E3–1230 CPU running at 3.40 GHz with 64 GB of RAM, UnclearBallot took an average of 279 ms to process each ballot. For reference, Hart's fastest central scanner's maximum scan rate is one ballot per 352 ms [203], well above the time needed to carry out our attack.

These results show that UnclearBallot can successfully and efficiently manipulate ballot images to change real voters' marks. Moreover, the alterations likely would be undetectable to human auditors who examined only the ballot images.

3.6 Discussion and Mitigations

UnclearBallot demonstrates the need for a software-independent evidence trail against which election results can be checked. It shows that audits based on software which is independent from the rest of the election system is still not software independent. To date, the only robust and secure election technology that is widely used is optical-scan paper ballots with risk-limiting audits based on a robust, well-maintained, *physical* audit trail. However, image audits are not useless, and here we discuss uses for them as well as potential mitigations for our attack.

3.6.1 Uses for image audits.

So long as image audits are not the sole mechanism for verifying election results, they do provide substantial benefits to election officials. Using an image audit vastly simplifies some functions of election administration, like ballot adjudication in cases where marks cannot be interpreted by scanners or are otherwise ambiguous. Image audits can be used to efficiently identify and document election discrepancies, as has occurred in Maryland where nearly 2,000 ballots were discovered missing from the audit trail in 2016 [153]. Image audits also identified a flaw in the ES&S DS850 high speed scanner, where it was causing some ballots to stick together and feed two at a time [154].

Another way to utilize image audits is a transitive audit. Methods like SOBA [30] seek to construct an audit trail using all available means of election evidence, rooting the audit in some verification of physical record. By using physical records to verify other records, like CVRs or ballot images, confidence in election outcomes can be transitively passed on to non-physical audit trails. The drawback with this kind of audit is that it usually requires the same level of work as an RLA, plus whatever work is needed to validate the other forms of evidence. However, since ballot image audits already require a low amount of effort, they may augment RLAs and provide better transparency into the auditing process.

Image audits are an augmentation and a convenience for election administration, however, and should not be viewed as a security tool. Only physical examination of paper ballots, as in a risk-limiting audit, can provide a necessary level of mitigation to manipulated election results.

3.6.2 End-to-end (E2E) systems.

Voting systems with rigorous integrity properties and tamper resistance such as Scantegrity [60] and Prêt à Voter [198] provide a defense to UnclearBallot. In Scantegrity, when individuals mark their ballots, a confirmation code is revealed that is tied to the selected candidate. This enables a voter to verify that their ballot is collected-as-cast and counted-as-collected, as they can look up their ballot on a public bulletin board. Since each mark reveals a unique code, moving the mark would match the code with the wrong candidate, so voters would be unable to verify their ballots. If enough voters complain, this might result in our attack being detected.

Prêt à Voter randomizes the candidate order on each ballot, which creates a slightly higher barrier for our attack, as an additional template matching step would be needed to ascertain candidate order. More importantly, the candidate list is physically separated from the voter's marks upon casting the ballot, so malware which could not keep track of the correct candidate order could not successfully move marks to a predetermined candidate. Since the candidate order is deciphered via a key-sharing scheme, malicious software would have to infect a significant portion of the election system and act in a highly coordinated way to reconstruct candidate ordering. Moreover, as with Scantegrity, votes are published to a public bulletin board, so any voter could discover if their vote had not been correctly recorded.

Other E2E systems which make use of optical scanning and a bulletin board, like STAR-Vote [26], Scratch and Vote [8], and VeriScan [29], are similarly protected from attacks like UnclearBallot.

3.6.3 Other mitigations.

Outside of E2E, there may be other heuristic mitigations that can be easily implemented even in deployed voting systems to make our attack somewhat more difficult. As mentioned above, randomizing candidate order on each ballot increases the computation required to perform our attack. Voters drawing outside the bubbles can also defeat our attack, though this might also result in their votes not counting and may be circumvented by replacing the whole race on the ballot image with a substituted one. Collecting ballot images from a different source than the tabulator makes our attack more difficult, as votes now have to be changed in two places. Other standard computer security technologies, like secure file systems, could be used to force the attacker to alter ballot images in a way that also circumvents protections like encryption and permissions.

3.6.4 Detection.

Technologies that detect image manipulation may also provide some mitigation. Techniques like those discussed in [21–23, 213], among others, could be adapted to try to automatically detect moved marks on ballots. However, as noted by Farid [97], image manipulation detection is a kind of arms race: given a fixed detection algorithm, adversaries can very likely find a way to defeat it. In our context, an attacker with sufficient access to the voting system to implant a manipulation algorithm would likely also be able to to steal the detector code. The attacker could improve the manipulation algorithm or simply use the detector as part of their mark-moving calculus: if moving a mark will trip the detector, an attacker can simply opt not to move the mark.

While a fixed and automatic procedure for detecting manipulation can provide little assurance, it remains possible that an adaptive approach to detection could be a useful part of a post-election forensics investigation. However, staying one step ahead of sophisticated adversaries would require an ongoing research program to advance the state of the art in detection methods.

A less costly and more dependable way to detect ballot manipulation detection would be to use a software independent audit trail to confirm election outcomes. This can be accomplished with risk-limiting audits, and the software independence enabled by RLAs provides other robust security properties to elections, including defending against other potential attacks on tabulation equipment and servers.

3.6.5 Future work.

We have only focused on simple-majority elections here, because those are the kinds of elections used by jurisdictions that do image audits. Audits of more complex election methods, like instant-runoff voting or D'Hondt, have been examined to some extent [202, 220], but future work is needed into audits of these kinds of elections altogether. Because the marks made in these elections are different than the kind we've discussed here, manipulation of these ballot images may not be able to employ the same image processing techniques we have used. Additionally it may be difficult for malware to know how many marks it needs to move, since margins in complex elections are difficult to compute. We leave exploration of image manipulation of these elections to future work.

3.7 Conclusion

In this chapter, we demonstrated an attack that defeats ballot image audits of the type performed in some jurisdictions. We presented an implementation using a real scanner, and evaluated our implementation against a set of real ballots and a set of systematically marked ballots from a variety of ballot styles. Our attack shows that image audits cannot be relied upon to verify that elections are free from computer-based interference. Indeed, the only currently known way to verify an election outcome is with direct examination of physical ballots. In order to secure an election, physical paper ballots are necessary.

We have also noted that the right election security tools, in this case auditing, can still go awry when their motivations and requirements are not met. Image audits do in a sense rely on paper ballots, but not *in the right way*. As we shall see in the next chapter, correctly using a paper ballot is even more nontrivial than just auditing the physical paper. Having paper and auditing it is not enough; ballots must also be voter-verified.

CHAPTER IV

Voter-verified Paper[1]

4.1 Introduction

As we have just seen, paper ballots are essential to providing robust security for elections. Recent threats of election hacking by hostile nations has prompted a major push to ensure that all voting systems in the United States have voter-verifiable paper trails, a move recommended by the National Academies [160], the Senate Select Committee on Intelligence [227], and nearly all election security experts. Guided by past research [26], some states and localities are implementing paper trails by deploying ballot-marking devices (BMDs). In these systems, the voter makes selections on a computer kiosk, which prints a paper ballot that the voter can review before inserting it into a computer scanner to be counted [233]. BMDs have long been used as assistive devices for voters with disabilities, and a growing number of jurisdictions are purchasing them for use by all voters [99, 101, 161].

BMDs have the potential to provide better security than direct-recording electronic voting machines (DREs), which maintain the primary record of the voter's selections in a computer database and often lack a voter-verifiable paper trail. Numerous studies have demonstrated vulnerabilities in DREs that could be exploited to change election results (e.g., [18, 48, 131, 155]). In contrast, BMDs produce a physical record of every vote that can, in principle, be verified by the voter and manually

[1] This chapter is based on "Can Voters Detect Malicious Manipulation of Ballot-marking Devices?", work in conjunction with Allison McDonald, Henry Meng, Jensen Hwa, Nakul Bajaj, Kevin Chang, and J. Alex Halderman that appeared in *Proceedings of the 41st IEEE Symposium on Security and Privacy* [37]

audited by officials to confirm or correct the initial electronic results.

However, BMDs do not eliminate the risk of vote-stealing attacks. Malware could infect the ballot scanners and change the electronic tallies—although this could be detected by rigorously auditing the paper ballots [215]—or it could infect the BMDs themselves and alter what gets printed on the ballots. This latter variety of cheating cannot be detected by a post-election audit, since the paper trail itself would be wrong, and it cannot be ruled out by pre-election or parallel testing, though those techniques may provide some mitigation [218, 242]. Instead, BMD security relies on voters themselves detecting such an attack. This type of human-in-the-loop security is necessary in many systems where detection and prevention of security hazards cannot be automated [82]. However, as several commentators have recently pointed out [15, 86, 218], its effectiveness in the context of BMDs has not been established.

Whether such a misprinting attack would succeed without detection is highly sensitive to how well voters verify their printed ballots. Every voter who notices that their ballot is misprinted and asks to correct it *both* adds to the evidence that there is a problem *and* requires the attacker to change an additional ballot in order to overcome the margin of victory. Consider a contest with a 1% margin in which each polling place has 1000 voters. If voters correct 20% of misprinted ballots, minimal outcome-changing fraud will result in an average of 1.25 voter complaints per polling place—likely too few to raise alarms. If, instead, voters correct 80% of misprinted ballots, polling places will see an average of 20 complaints, potentially prompting an investigation. (We model these effects in Section 4.5.) Despite this sensitivity, voters' BMD verification performance has never before been experimentally measured.

In this chapter, we study whether voters can play a role in BMD security. We first seek to establish, in a realistic polling place environment, the rates at which voters attempt to verify their printed ballots and successfully detect and report malicious changes. To measure these rates, we used real touch-screen voting machines that we modified to operate as malicious BMDs. We recruited 241 participants in Ann Arbor, Michigan, and had them vote in a realistic mock polling place using the ballot from the city's recent midterm election. On every ballot that our BMDs printed, one race was changed so the printout did not reflect the selection made by the participant.

We found that, absent interventions, only 40% of participants reviewed their printed ballots at all, only 6.6% reported the error to a poll worker, and only 7.8% correctly identified the error on an exit survey. These results accord with prior studies that found poor voter performance in other election security contexts, such as DRE review screens [6, 58] and voter-verifiable paper audit trails (VVPATs) [205]. The low rate of error detection indicates that misprinting attacks on BMDs pose a serious risk.

The risks notwithstanding, BMDs do offer practical advantages compared to hand-marked paper ballots. They allow voters of all abilities to vote in the same manner, provide a more user-friendly interface for voting, and more easily support complex elections like those conducted in multiple languages or with methods such as ranked choice [185]. BMDs also simplify election administration in places that use vote centers [233], which have been shown to reduce election costs and lower provisional voting rates [120, 181], as well as in jurisdictions that employ early voting, which can improve access to the ballot [127].

Given these advantages and the fact that BMDs are already in use, the second goal of our study was to determine whether it might be possible to boost verification performance through procedural changes. We tested a wide range of interventions, such as poll worker direction, instructional signage, and usage of a written slate of choices provided to each voter.

The rate of error detection varied widely with the type of intervention we applied, ranging from 6.7% to 86% in different experiments. Several interventions boosted review rates and discrepancy reporting. Verbally encouraging participants to review their printed ballot after voting boosted the detection rate to 14% on average. Using post-voting verbal instructions while encouraging participants to vote a provided list of candidates raised the rate at which voters reported problems to 73% for voters who did not deviate from the provided slate.

These findings suggest that well designed procedures can have a sizable impact on the real-world effectiveness of voter verification. We make several recommendations that election officials who already oversee voting on BMDs can employ immediately, including asking voters if they have reviewed their ballots before submission, promoting the use of slates during the voting process, informing voters that if they find an error in the printout they can correct it, and tracking the rate of reported errors. Our recommendations echo similar findings about the most effective ways to alert users to other security hazards (i.e., in context [51] and with active alerts [92]) and redirect them to take action.

Although our findings may be encouraging, we strongly caution that much additional research is necessary before it can be concluded that any combination of procedures actually achieves high verification performance in real elections. Until BMDs are shown to be effectively verifiable during real-world use, the safest course for security is to prefer hand-marked paper ballots.

Road Map Section 4.2 provides more background about human factors and security and about previous work studying the role of voter verification in election security. Section 4.3 describes our experimental setup, voting equipment, and study design. Section 4.4 presents our results and analyzes their significance. Section 4.5 provides a quantitative model for BMD verification security. Section 4.6 discusses the results, avenues for future work, and recommendations for improving the verifiability of BMDs. We conclude in Section 4.7.

4.2 Background and Related Work

4.2.1 Human-Dependent Security

Secret ballot elections fundamentally depend on having humans in the loop—as Stark [218] notes, the voter is the *only one* who knows whether the ballot represents their intended vote—and the success or failure of election security has the potential to have history-altering effects. The type of risk posited by Stark, wherein voters do not check their paper ballots to ensure the BMD has correctly represented their selections, is a post-completion error [55], in which a user makes a mistake (or fails to verify the correctness of something) *after* they have completed the main goal of their task. Voters who forget or do not know to verify the correctness of a paper ballot after they have entered their selections on a BMD miss a critical step in ensuring the accuracy of their vote: that it was cast-as-intended. We therefore explore how to communicate this risk to voters.

Cranor [82] describes five ways that designers can communicate risk to a user who needs to make security decisions:

1. *Warnings*: indication the user should take immediate action

2. *Notices*: information to allow the user to make a decision

3. *Status indicators*: indication of the status of the system

4. *Training*: informing users about risks and mitigations before interaction

5. *Policies*: rules with which users are expected to comply

Implementing indicators that reveal meaningful information to voters about the security status of a BMD would be next to impossible, as security issues are often unknown or unforeseen to the operators. Although voter education about the importance of verification might be an effective form of training, significant coordination would be necessary to enact such a scheme at scale. Therefore, we focus in this study on the effectiveness of warnings issued through poll worker scripts and polling place signage.

A warning serves two purposes: to alert users to a hazard, and to change their behavior to account for the hazard [246]. There are many barriers to humans correctly and completely heeding security warnings. Wogalter proposes the Communication-Human Information Processing (C-HIP) Model [245] to systematically identify the process an individual must go through for a warning to be effective. The warning must capture and maintain attention, which may be difficult for voters who are attempting to navigate the voting process as quickly as possible. Warnings must also be comprehensible, communicate the risks and consequences, be consistent with the individual's beliefs and attitudes toward the risk, and motivate the individual to change—all of which are substantial impediments in an environment with little to no user training and such a broad user base as voting.

To maximize effectiveness, warnings should be contextual, containing as little information as necessary to convey the risk and direct individuals to correct behavior [51, 245]. Voters are essentially election security novices; Bravo-Lillo et al. [51] found that, in the context of computer security, advanced and novice users respond to warnings differently. Most significantly, novice users assessed the hazard *after* taking action, whereas advanced users assessed the hazard *before* engaging in the activity.

There may be effective ways to improve voter verification performance. Many studies have applied lessons from Cranor, Wogalter, and Bravo-Lillo et al. to help humans make secure choices in different contexts, including phishing [92, 180], browser warnings [9, 187, 221], app permissions [10, 179], and operating system interfaces [52]. In the context of phishing warnings, for example, Egelman et al. [92] found that users were far more likely to heed an active warning, or a warning that disrupted their workflow, than a passive warning. This suggests that similar interventions applied in a polling place may have a significant effect on voters' ability to review and verify their BMD ballots.

Our study contributes to this literature by exploring the effects of several modalities of warnings (oral and visual) on human detection of malicious ballot modification.

4.2.2 Usable Voting Systems

The usability of various kinds of election technology has been extensively studied, primarily to determine how well voters can use voting equipment. Olembo and Volkamer [174] provide an excellent overview of the space, along with recommendations for designing voting usability studies. Quesenbery [184] also provided earlier guidelines for usability studies of voting equipment.

In 2003, Bederson et al. [24] studied the usability of deployed electronic voting systems in Maryland via expert review and field study. Conrad et al. [78] later performed laboratory experiments with six models of voting equipment. In the last decade, Byrne et al. performed multiple experiments establishing baseline data for various types of voting systems [56], as did Greene et al. [106]. Everett et al. assessed the usability of paper ballots [95] and electronic voting [96]. Herrnson et al. also performed extensive analysis of the usability of electronic voting systems [116–119].

Most of these studies focus on in-precinct voting systems, which require a voter to vote in-person. In addition, there are numerous works on the usability of end-to-end cryptographic voting schemes [2, 3, 5] and Internet voting schemes [87, 150, 151]. Acemyan et al. [4] also performed usability studies of other parts of the voting process, like precinct layout, and Belton et al. [27] examined the usability of ballot boxes. Acemyan et al. [5] examined the usability of STAR-Vote [26], an academic effort to produce a secure and usable BMD.

4.2.3 Voter-Verifiable Paper and Ballot-Marking Devices

A guiding principle in election security is that voting systems should be *software independent* [194]: that is, any software errors or attacks that change the reported election outcome should be detectable. Bernhard et al. [35] note that elections backed by a voter-verifiable paper record are currently the only known way to provide robust software independence. Like BMDs, voter-verifiable paper audit trails (VVPATs) and hand-marked paper ballots are widely used in an attempt to achieve software independence. However, each poses a different set of usability and accessibility challenges.

Hand-marked paper ballots record the voter's selections without the risk of having a potentially compromised computer mediating the process. However, voters can make mistakes when filling out ballots by hand that can lead to them being counted incorrectly or ruled invalid [106]. Moreover, many voters have difficulty marking a paper ballot by hand due to a disability or a language barrier. Ballots in the U.S. are among the most complex in the world, further magnifying these difficulties [164].

VVPAT technology also suffers from noted usability, privacy, and auditability problems [104]. Most implementations consist of clunky printer attachments for DREs that are difficult for voters to read, record votes in the order in which they are cast, and use a fragile paper tape. In laboratory studies, Selker et al. [205] and de Jong et al. [84] found that voters frequently did not review the VVPAT, with Selker finding that only 17% of voters detected changes between the selections they made on the DRE and those printed on the VVPAT. While there has been some criticism of Selker's findings and methodology [186, 206], their results broadly comport with work by Campbell et al. [58] and Acemyan et al. [6] about voters' ability to detect errors introduced in DRE review screens. The latter found that only 12–40% of participants successfully detected such errors.

In part due to the concerns raised by these studies, BMDs have become a popular choice for new voting system deployments in the United States. South Carolina and Georgia, together comprising nearly 9 million voters, recently adopted BMDs statewide [99, 101], as have several counties and cities, including Los Angeles County, the largest single election jurisdiction in the U.S. [238].

There has been vigorous debate among election security experts as to whether BMDs can provide software-independence and election security overall (e.g., [15, 86, 218, 242]). However, the discussion has yet to be informed by rigorous experimental data. Our work seeks to fill that gap by contributing the first human-subjects study to directly measure the verification performance of voters using BMDs under realistic conditions and with a variety of potential procedural interventions.

After our work was published, Kortum, Byrne, and Whitmore publish results from a similar study [132]. Their methodology differed in their experimental setup, as they relied on a laboratory setting rather than a precinct-like environment we attempted to create. The ballot design and voting interfaced used by Kortum et al. is likely more usable than our setup, as it was based on more modern technology developed for L.A. County's VSAP project [238], though they also tested a ballot styled after the ES&S ExpressVote system, which may be less usable than the ballots we tested. They examined a much longer ballot than ours, with 40 races, as well as a shorter ballot with just five. Kortum et al. also varied the number of changes they made, ranging from one flip like our study to flipping 40% of the ballot, as well as which part of the ballot the flips were located. They also examined two different instruction sets for voters. Their results largely comport with ours, and I note the relevant details below as applicable.

4.3 Materials and Methods

Our goals in this work were to empirically assess how well voters verify BMD ballots and whether there are steps election officials can take that will enhance verification performance. To these ends, we conducted a between-subjects study where we tested several hypotheses in a simulated polling place, following the best practices recommended by Olembo et al. [174] for election human-factors research. The study design was approved by our IRB.

We sought to answer several questions, all of which concern the rate at which voters are able to detect that a BMD-printed ballot shows different selections than those the voter picked:

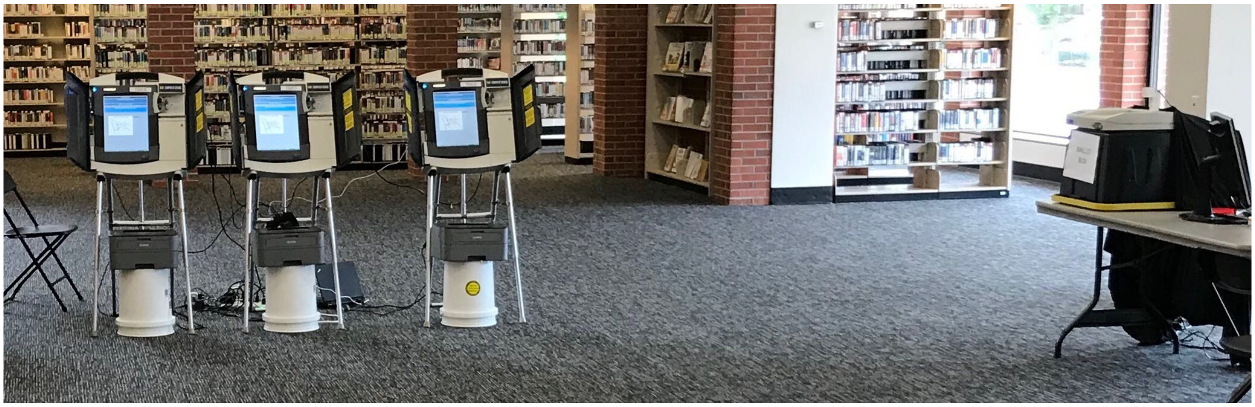


Figure 4.1: *Polling Place Setup.* We established mock polling places at two public libraries in Ann Arbor, Michigan, with three BMDs (*left*) and an optical scanner and ballot box (*right*). Library visitors were invited to participate in a study about a new kind of election technology. The BMDs were DRE voting machines that we modified to function as malicious ballot marking devices.

- What is the base rate of error detection?
- Is error detection impacted by:
  - Ballot style?
  - Manipulation strategy?
  - The manipulated race's position on the ballot?
  - Signage instructing voters to review their ballots?
  - Poll worker instructions?
  - Providing a slate of candidates for whom to vote?

In order to answer these questions in an ecologically valid way, we attempted to create an environment that closely resembled a real polling place. Nevertheless, it is impossible for any experiment to fully recreate what is at stake for voters in a real election, and so study participants may have behaved differently than voters do in live election settings. We went to extensive lengths to mitigate this limitation, and we find some data to support that we did so successfully (see Section 4.6.1). We used real (though modified) voting machines, printers and paper stock from deployed BMD systems, a ballot from a real election, and ballot styles from two models of BMDs.

We conducted the study at two city library locations, one of which is used as a polling place during real elections.

4.3.1 The Polling Place

To provide a realistic voting experience, we structured our simulated polling place like a typical BMD-based poll site. Three investigators served as poll workers, following the script in Appendix 1.1. Library patrons who were interested in voting began at a check-in table, where they were greeted by Poll Worker A and asked to sign an IRB-approved consent form. Participants were told they would be taking part in "a study about the usability of a new type of voting machine" and instructed on how to use the equipment, but they were not alerted that the study concerned security or that the BMDs might malfunction.

Each participant received a voter access card with which to activate a BMD and was free to choose any unoccupied machine. There were three identical BMDs, as shown in Figure 4.1. On the last day of the study, one machine's memory became corrupted, and it was removed from service; all votes that day were recorded on the other two machines.

The BMDs displayed contests in a fixed order, and voters made selections using a touch screen interface. After the last contest, the machines showed a review screen that accurately summarized the voter's selections and highlighted any undervotes. The voter could return to any contest to change the selections. A "Print Ballot" button ended the voting session and caused a printer under the machine to output the paper ballot.

Participants carried their ballot across the polling place to the ballot scanner station, where they inserted them into an optical scanner that deposited them into a ballot box. Poll Worker B was stationed by the scanner and offered instructions if necessary. Next, the poll worker collected the voter access card and asked each participant to complete an exit survey using a laptop next to the scanning station. The survey was anonymous, but responses were keyed so that we could associate them with the voter's on-screen selections, their printed ballot, and poll worker notes.

Poll Worker C, positioned separately from the other stations, acted as an observer. They verified that participants moved through the polling place stations sequentially, noted whether they spent time reviewing their printed ballots, and recorded whether they appeared to notice any abnormalities. The observer was also tasked with noting participant behavior, specifically how the participants completed each step in the voting process and any comments they made. The observer was available to answer participant questions and was frequently the poll worker participants approached upon noticing a discrepancy.

Like in a real polling place, multiple participants could progress through the voting process simultaneously. Occasionally a one- or two-person line formed as participants waited to use the BMDs or the ballot scanner.

4.3.2 The Voting Machines

BMD voting systems are currently produced by several voting machine manufacturers, the largest of which is ES&S. Over a six month period, we repeatedly attempted to engage ES&S in discussions about acquiring samples of their equipment for this study. However, these attempts were ultimately not fruitful.

Instead, we utilized AccuVote TSX DRE voting machines, which we purchased on eBay and modified to function as BMDs. The TSX was first produced by Diebold in 2003 and is still widely deployed today. At least 15 states plan to use it in at least some jurisdictions in November 2020 [234].

The TSX runs Windows CE and is designed to function as a paperless DRE or a VVPAT system. We developed software modifications that allow it to print ballots in multiple styles using an external printer. This effectively converts the TSX into a BMD—and one we could easily cause to be dishonest—while preserving the original touch-screen interface used by voters.

In order to modify the machine, we built on techniques used by Feldman et al. [18]. We began by patching the firmware so that, when the machine boots, it attempts to execute a program provided on an external memory card. We used this functionality to launch a remote access tool we created, which allowed us to connect to the TSX over a network and perform file system operations, run applications, and invoke a debugger.

The TSXes in our polling place were connected to an Ethernet switch using PCMCIA network adapters. A Python program, running on a computer on the same network, used the remote access tool's API to poll each machine for newly voted ballots. Whenever a ballot was cast, the program parsed the selections, generated a PDF file based on them, and sent it to a printer located underneath the appropriate voting machine. The program could be configured to apply different ballot styles and cheating strategies, depending on the experiment.

For every ballot, the program randomly selected one race to manipulate. In most experiments, selections could be changed in three ways: deselection in a voted-for race, selection in an unvoted-for race, or changing a selection to a different candidate. We ensured that some alteration would take place on every ballot. For example, in a vote-for-one race where the voter had made a selection, the algorithm would choose uniformly from the set of unselected choices plus no selection. One experiment used a different strategy, in which choices could only be deselected.

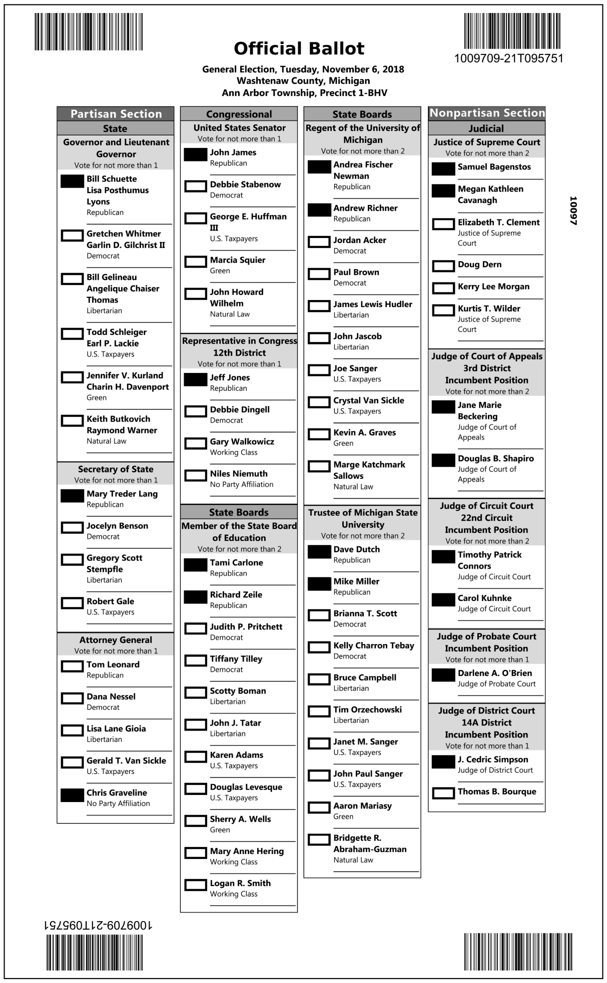
Both the voter's original selections and the manipulated ballot were logged for later analysis. Each voting session was associated with a unique tracking number, which was printed on the ballot along with a timestamp and encoded as a barcode.

As the final step in the voting process, participants fed their printed ballots into an AccuVote OS optical scanner, a device used to tabulate votes in parts of 20 states [234]. The scanner was intended to add realism to the experiment, but AccuVote OSes are not capable of actually tabulating the ballot styles we used. Therefore, we modified the scanner so that it simply fed each ballot into the ballot box without counting it.

We mounted a barcode reader in a 3-D printed case above the scanner's input tray and positioned it so that it would detect the ballot's tracking barcode. (This setup can be seen in Figure 4.3.) When the barcode was read, a Raspberry Pi would activate the AccuVote OS's feed motor to pull the ballot into the ballot box. The Raspberry Pi also displayed the ballot tracking number so that poll workers could associate the ballot with the participant's exit survey response and the observer's notes.

4.3.3 The Ballot

In order to ensure a realistic voting experience and increase participants' psychological investment in the outcome of the mock election, we used races and candidates from the city's actual ballot



Official Ballot

1009709-21T095751

General Election, Tuesday, November 6, 2018
Washtenaw County, Michigan
Ann Arbor Township, Precinct 1-BHV

Partisan Section

State

Governor and Lieutenant Governor
Vote for not more than 1

■ **Bill Schuette**
Lisa Posthumus Lyons Republican
□ **Gretchen Whitmer**
Garlin D. Gilchrist II Democrat
□ **Bill Gelineau**
Angelique Chaiser Thomas Libertarian
□ **Todd Schleiger**
Earl P. Lackie U.S. Taxpayers
□ **Jennifer V. Kurland**
Charin H. Davenport Green
□ **Keith Butkovich**
Raymond Warner Natural Law

Secretary of State
Vote for not more than 1

■ **Mary Treder Lang** Republican
□ **Jocelyn Benson** Democrat
□ **Gregory Scott Stempfle** Libertarian
□ **Robert Gale** U.S. Taxpayers

Attorney General
Vote for not more than 1

□ **Tom Leonard** Republican
□ **Dana Nessel** Democrat
□ **Lisa Lane Gioia** Libertarian
□ **Gerald T. Van Sickle** U.S. Taxpayers
■ **Chris Graveline** No Party Affiliation

Congressional

United States Senator
Vote for not more than 1

■ **John James** Republican
□ **Debbie Stabenow** Democrat
□ **George E. Huffman III** U.S. Taxpayers
□ **Marcia Squier** Green
□ **John Howard Wilhelm** Natural Law

Representative in Congress 12th District
Vote for not more than 1

■ **Jeff Jones** Republican
□ **Debbie Dingell** Democrat
□ **Gary Walkowicz** Working Class
□ **Niles Niemuth** No Party Affiliation

State Boards

Member of the State Board of Education
Vote for not more than 2

■ **Tami Carlone** Republican
■ **Richard Zeile** Republican
□ **Judith P. Pritchett** Democrat
□ **Tiffany Tilley** Democrat
□ **Scotty Boman** Libertarian
□ **John J. Tatar** Libertarian
□ **Karen Adams** U.S. Taxpayers
□ **Douglas Levesque** U.S. Taxpayers
□ **Sherry A. Wells** Green
□ **Mary Anne Hering** Working Class
□ **Logan R. Smith** Working Class

State Boards

Regent of the University of Michigan
Vote for not more than 2

■ **Andrea Fischer Newman** Republican
■ **Andrew Richner** Republican
□ **Jordan Acker** Democrat
□ **Paul Brown** Democrat
□ **James Lewis Hudler** Libertarian
□ **John Jascob** Libertarian
□ **Joe Sanger** U.S. Taxpayers
□ **Crystal Van Sickle** U.S. Taxpayers
□ **Kevin A. Graves** Green
□ **Marge Katchmark Sallows** Natural Law

Trustee of Michigan State University
Vote for not more than 2

■ **Dave Dutch** Republican
■ **Mike Miller** Republican
□ **Brianna T. Scott** Democrat
□ **Kelly Charron Tebay** Democrat
□ **Bruce Campbell** Libertarian
□ **Tim Orzechowski** Libertarian
□ **Janet M. Sanger** U.S. Taxpayers
□ **John Paul Sanger** U.S. Taxpayers
□ **Aaron Mariasy** Green
□ **Bridgette R. Abraham-Guzman** Natural Law

Nonpartisan Section

Judicial

Justice of Supreme Court
Vote for not more than 2

■ **Samuel Bagenstos**
■ **Megan Kathleen Cavanagh**
□ **Elizabeth T. Clement** Justice of Supreme Court
□ **Doug Dern**
□ **Kerry Lee Morgan**
□ **Kurtis T. Wilder** Justice of Supreme Court

Judge of Court of Appeals 3rd District Incumbent Position
Vote for not more than 2

■ **Jane Marie Beckering** Judge of Court of Appeals
■ **Douglas B. Shapiro** Judge of Court of Appeals

Judge of Circuit Court 22nd Circuit Incumbent Position
Vote for not more than 2

■ **Timothy Patrick Connors** Judge of Circuit Court
■ **Carol Kuhnke** Judge of Circuit Court

Judge of Probate Court Incumbent Position
Vote for not more than 1

■ **Darlene A. O'Brien** Judge of Probate Court

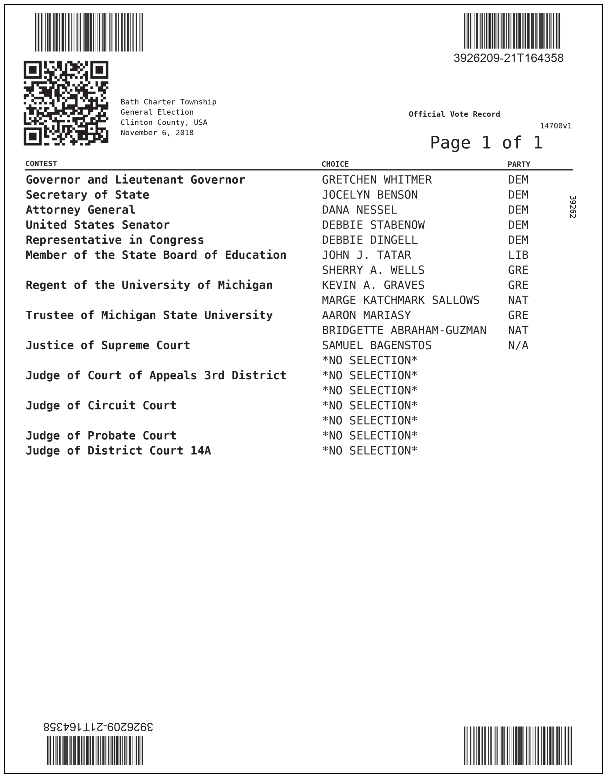
Judge of District Court 14A District Incumbent Position
Vote for not more than 1

■ **J. Cedric Simpson** Judge of District Court
□ **Thomas B. Bourque**

10097

1009709-21T095751

(a) Regular Ballot



3926209-21T164358

Bath Charter Township
General Election
Clinton County, USA
November 6, 2018

Official Vote Record

14700v1

Page 1 of 1

CONTEST	CHOICE	PARTY
Governor and Lieutenant Governor	GRETCHEN WHITMER	DEM
Secretary of State	JOCELYN BENSON	DEM
Attorney General	DANA NESSEL	DEM
United States Senator	DEBBIE STABENOW	DEM
Representative in Congress	DEBBIE DINGELL	DEM
Member of the State Board of Education	JOHN J. TATAR	LIB
	SHERRY A. WELLS	GRE
Regent of the University of Michigan	KEVIN A. GRAVES	GRE
	MARGE KATCHMARK SALLOWS	NAT
Trustee of Michigan State University	AARON MARIASY	GRE
	BRIDGETTE ABRAHAM-GUZMAN	NAT
Justice of Supreme Court	SAMUEL BAGENSTOS	N/A
	NO SELECTION	
Judge of Court of Appeals 3rd District	*NO SELECTION*	
	NO SELECTION	
Judge of Circuit Court	*NO SELECTION*	
	NO SELECTION	
Judge of Probate Court	*NO SELECTION*	
Judge of District Court 14A	*NO SELECTION*	

39262

3926209-21T164358

(b) Summary Ballot

Figure 4.2: *Ballot Styles.* We tested two ballot styles: (*a*) a regular style, resembling a hand-marked ballot; and (*b*) a summary style, listing only the selected candidates. Both had 13 races from the city's recent midterm election. In one race, determined randomly, the printed selection differed from the voter's choice.

for the recent 2018 midterm election. For simplicity, we reduced the ballot to the first 13 races so that ballots would not require duplex printing or multiple pages.

We tested two ballot styles, which are illustrated in Figure 4.2. One is a regular ballot that shows the entire set of candidates in every race. The other is a summary ballot, which shows only the voter's selections or "NO SELECTION" if a choice is left blank. Most BMDs print ballots that resemble these styles.

The specific visual designs we used mimic ballots produced by two models of BMDs manufactured by Hart InterCivic, which also makes the voting equipment used in Ann Arbor. The regular style is also the same design as the hand-marked paper ballots most Ann Arbor voters use, ensuring that many participants found it familiar. These designs are used in jurisdictions that collectively have over 10 million registered voters [234].

The model of laser printer we used, Brother HL-2340, is certified for use with Clear Ballot's ClearAccess BMD system [183], so we chose paper stock that meets the specifications for ClearAccess [74]. Summary ballots were printed on regular weight 8.5×11 inch letter paper, while regular ballots were printed on Vellum Bristol stock 67 pound 8.5×14 inch paper.

4.3.4 Participants and Recruitment

To gather subjects for our study, we approached staff at the Ann Arbor District Library (AADL), who offered space for us to set up our mock precinct. We conducted a total of three days of data collection in July and September 2019 at two library locations: the Downtown and Westgate branches. The Downtown branch, where our study was held for two of the three days, is an official polling location during real elections.

The AADL advertised our study through its social media feeds and offered incentives to patrons for their participation, such as points for a scavenger hunt competition [13] and souvenir flashlights [14]. We also set up a fourth voting machine outside of the mock precinct where kids could vote in an election for mayor of the library's fish tank.[2] Results from that machine were not

[2] Mighty Trisha unexpectedly beat Creepy Bob, leading some Bob supporters to complain that the results were fishy [12].

used as part of this study, but it served as a recruitment tool for parents visiting the library with their children. In addition, we verbally recruited patrons who happened to be at the libraries during our study, using the script in Appendix 1.2.

Participants were required to be at least 18 years of age and to sign an IRB-approved consent form. All data collected, including survey responses and behavioral observations, was completely anonymous. We informed participants that they were not required to vote their political preferences.

4.3.5 Experiments

To explore what factors affect voter verification performance, we devised nine experiments to run between subjects. In all experiments, for every participant, one selection that the participant made on the BMD was not accurately reflected on the printed ballot. Every participant within an experiment received the same instructions from the poll workers, following the script and variants in Appendix 1.1.

The first three experiments were designed to measure verification in the absence of protective interventions. They varied the ballot style and manipulation strategy:

E1: Regular ballots We used the regular ballot style and the default manipulation strategy, in which a selection could be switched, deselected, or selected if left blank by the voter.

E2: Summary ballots We used the summary ballot style and the default manipulation strategy. As discussed in Section 4.4, we found no significant difference in error detection between regular ballots and summary ballots, so all subsequent experiments used summary ballots.

E3: Deselection only To assess the sensitivity of voters to the way their ballots were changed, we limited the manipulation to deselecting one of the voter's choices at random.

Four further experiments tested interventions to determine if they improved error detection. We tried posting a sign and having poll workers give different instructions at various times:

E4: Signage A sign was placed above the scanner that instructed voters to check their printed ballots, as shown in Figure 4.3. We designed the sign following guidelines from the U.S. Election Assistance Commission [230].

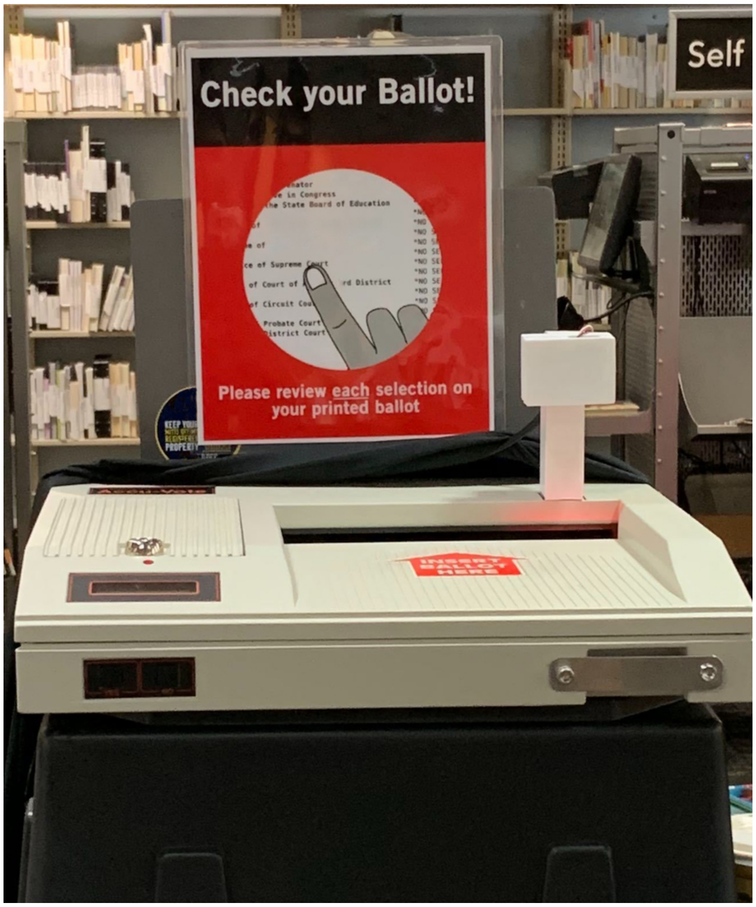


Figure 4.3: *Warning Signage.* One of the interventions we tested was placing a sign above the scanner that instructed voters to verify their ballots. Signage was not an effective intervention.

E5: Script variant 1 During voter check in, the poll worker added this instruction: "Please remember to check your ballot carefully before depositing it into the scanner."

E6: Script variant 2 When the voter approached the scanner, the poll worker said: "Please keep in mind that the paper ballot is the official record of your vote."

E7: Script variant 3 When the voter approached the scanner, the poll worker said: "Have you carefully reviewed each selection on your printed ballot?"

The final two experiments assessed whether reminding participants of their selections during verification improved their performance. We gave voters a slate of candidates for whom to vote that they could carry with them throughout the voting experience. While we refer to this as a slate, a sample ballot that the voter filled in before voting could serve the same purpose. Every voter received the same slate (Appendix 1.3), which was randomly generated and contained an even mix of parties.

E8: Slate with script variant 2 Voters were given the slate. Poll workers encouraged verification with script variant 2.

E9: Slate with script variant 3 Voters were given the slate. Poll workers encouraged verification with script variant 3.

4.4 Results

4.4.1 Participant Demographics

We recruited 241 participants. Their demographics are shown in Figure 4.4. The vast majority (220, 91%) indicated that they were native English speakers; 19 reported speaking twelve other native languages, including Hungarian, Korean, and Arabic; and two subjects gave no response. Participants who disclosed their age ranged from 18 to 84 years old, with a mean of 43.7 and a median of 42; 15 subjects did not answer the question. The percentages that follow are out of the total number of responses to each question: Respondents identified as male (84, 35%), female (152, 64%), or other (3, 1%); two did not respond. Subjects reported their ethnicity as Caucasian (187, 80%), Asian (17, 7%), African American (6, 3%), Mexican American/Chicano (5, 2%), and Other Hispanic/Latino (9, 4%); others reported not having any of these ethnic backgrounds (2, 1%) or were multiracial (9, 4%). Participants reported their level of educational attainment as some high school (1, 0.4%), a high school diploma (4, 2%), some college (20, 8%), a two-year degree (10, 4%), a four-year degree (80, 33%), a master's or professional degree (92, 38%), or a doctorate (34, 14%).

Most subjects indicated that they were registered to vote in the U.S. (220, 92%), had voted in a previous election (216, 91%), and had voted in the November 2018 midterm election (209, 87%). However, we note that, historically, 38–45% of non-voters have been found to falsely report having voted [38].

Compared to the population of Ann Arbor at the time of the 2010 census, our participant pool

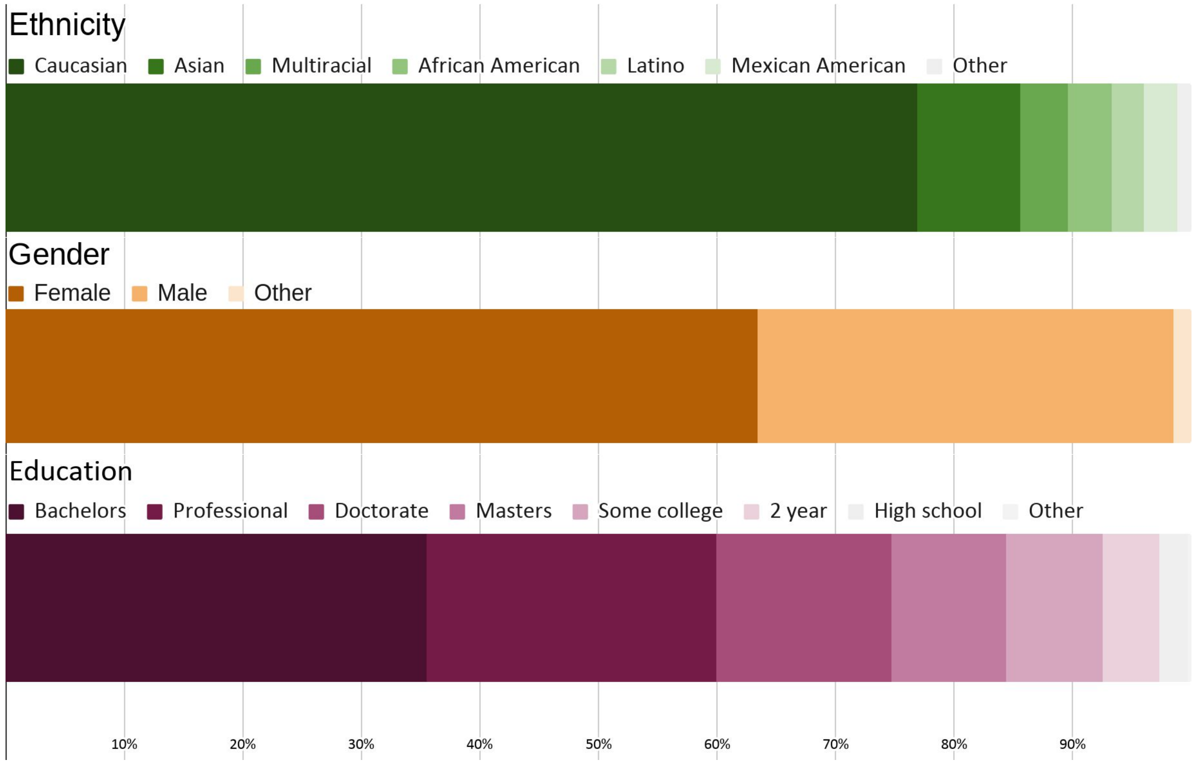


Figure 4.4: *Participant Demographics* Our participants largely reflected the demographics of Ann Arbor: they were well educated, mostly white, and mostly women.

overrepresented Caucasians ($\Delta = 7.6\%$) and underrepresented African Americans ($\Delta = -4.4\%$) and Asians ($\Delta = -8.7\%$) [229]. The study population also overrepresented females ($\Delta = 13\%$) and underrepresented males ($\Delta = -16\%$) [240]. In other reported aspects, participants' demographics resembled the population of Ann Arbor voters (the city is among the most highly educated in the U.S.) [147].

4.4.2 Verification Performance

To quantify verification performance, we collected three data points for each participant, which are summarized in Table 4.1. First, an observer noted whether the subject appeared to examine the printed ballot for at least two seconds. Second, the exit survey asked, "Did you notice anything odd about your ballot?", and we recorded whether the subject's response corroborated the discrepancy (i.e., correctly articulated which race was changed). Third, we recorded whether subjects reported the ballot modification to a poll worker. Most experiments saw more participants identify discrepancies

Experiment	*N*	Were observed examining ballot	Reported error on exit survey	Reported error to poll worker
Without interventions:				
E1: Regular ballots	31	41.9%	6.5%	6.5%
E2: Summary ballots	31	32.3%	6.5%	6.5%
E3: Deselection only	29	44.8%	10.3%	6.9%
Subtotal/Mean	91	39.7%	7.8%	6.6%
With interventions:				
E4: Signage	30	13.3%	3.3%	6.7%
E5: Script variant 1	30	46.7%	13.3%	6.7%
E6: Script variant 2	25	92.0%	16.0%	16.0%
E7: Script variant 3	31	38.7%	19.4%	12.9%
E8: Slate with script variant 2	13	100.0%	38.5%	38.5%
E9: Slate with script variant 3	21	95.2%	71.4%	85.7%
Subtotal/Mean	150	64.3%	24.0%	27.8%

Table 4.1: *Verification Performance for Each Experiment.* Without interventions, participants' verification performance was remarkably poor: only 7.8% noted on an exit survey that their ballots had been altered, and only 6.6% informed a poll worker (averaged across experiments). The various interventions we tested had widely different effects, ranging from no significant improvement (**E4**, **E5**) to a large increase in verification success (**E8**, **E9**).

in the survey than were reported to poll workers, but these differences were not statistically significant. Where applicable, we refer to participants who by some means reported detecting the discrepancies as "noticers" and those who did not as "non-noticers".

4.4.2.1 Performance without interventions (E1–E3)

With no interventions, we found verification performance to be consistently poor. The three experiments involved 91 participants, and, averaged across the experiments, only 40% of participants examined their ballots, only 7.8% noted the error on the exit survey, and only 6.6% reported it to a poll worker. We did not find significant differences in performance between regular and summary ballots or between the tested attack strategies.

4.4.2.2 Effectiveness of interventions (E4–E9)

The tested interventions resulted in a wide range of effect sizes. Neither signage (**E4**) nor poll worker instructions issued before the participant began voting (**E5**) yielded a statistically significant improvement to any aspect of verification performance. In contrast, poll worker instructions issued *after* the ballot was printed (**E6** and **E7**) did have a positive effect, boosting reporting rates to 20% on the exit survey and 14% to poll workers (averaged across the experiments).

The largest performance gains occurred when participants were directed to vote using a slate of candidates (**E8** and **E9**). However, only **E9** produced a statistically significant difference in reporting rates (Fisher's exact $p < 0.001$).[3] Averaged across both experiments, reporting rates increased to 55% on the exit survey and 62% to poll workers. **E8**, in which participants were directed how to vote using a slate of candidates, saw detection and reporting rates of 39%, which is similar to results for DRE review screen performance found by Campbell et al. [58] and Acemyan et al. [6], in studies that similarly directed participants how to vote. With script variant 3, the use of a slate produced a significant difference (comparing **E7** and **E9**, Fisher's exact $p < 0.02$) for both review and report, but it did not produce a significant difference using script variant 2 (comparing **E6** and **E8**). This indicates that voters may be sensitive to the specific instructions they receive about reviewing their ballots.

4.4.3 Correlates

4.4.3.1 Reviewing the ballot

Reviewing the ballot at all was significantly correlated with error reporting (two-sample permutation test $p < 0.001$ with 10k repetitions). Some interventions do seem to promote reviewing: **E6**, **E8**, and **E9** saw significant increases (Fisher's exact $p < 0.004$), although **E7** did not.

This seems to support the notion that voters are capable of detecting errors *if they review their ballot*. Indeed, while our baseline rate was only 7%, 19% of voters who reviewed their ballot detected problems in the non-interventions conditions. 41% of participants who were observed

[3] All p-values were computed with a Bonferroni correction at a family-wise error rate of 0.05.

reviewing their ballots in our intervention conditions detected issues, and this comports with forthcoming work by Kortum et al. who found similar results [132] with a different sample. While this observation may appear trite, it does directly refute an earlier hypothesis about the cognitive abilities of voters [86].

4.4.3.2 Time to ballot submission

Careful verification takes time, so one might expect that participants who noticed discrepancies took more time to cast their ballots. As an upper bound on how long subjects spent verifying, we calculated the time from ballot printing to ballot submission. (Due to clock drift on one of our machines, data from the third day of experiments was unusable, and consequently **E4** and **E7** are excluded from our timing analysis.) As expected, we find that noticers took an average of 121 s between printing and ballot submission (median 114 s), compared to only 43 s for non-noticers (median 32 s). This difference is statistically significant (two-sample permutation test $p < 0.004$, 10k iterations).

We compared the submission times for two sets of experiments: ones with extra instructions to the voter (**E5**, **E6**, **E8**, and **E9**; $N = 84$) and ones without (**E1**, **E2**, and **E3**; $N = 91$). The distributions of submission times are shown in Figure 4.5. The experiments that asked participants to review their ballots saw significantly more time spent between ballot printing and submission (two-sample permutation test $p < 0.004$, 10k iterations), an average of 83 s (median 72 s) compared to 50 s without (median 33 s).

Notably, participants who were given a slate of candidates to vote for had much higher submission times (two-sample permutation test $p < 0.004$, 10k iterations). Noticers in the slate experiments took an average of 119 s (median 111 s) and non-noticers averaged 55 s (median 52 s). This might be partly attributed to voters having to select unfamiliar candidates and wanting to check their work.

4.4.3.3 Demographics

Comparisons of detection rates across demographic groups revealed that a strong indicator for verification performance was voting experience. Subjects who reported being registered to vote

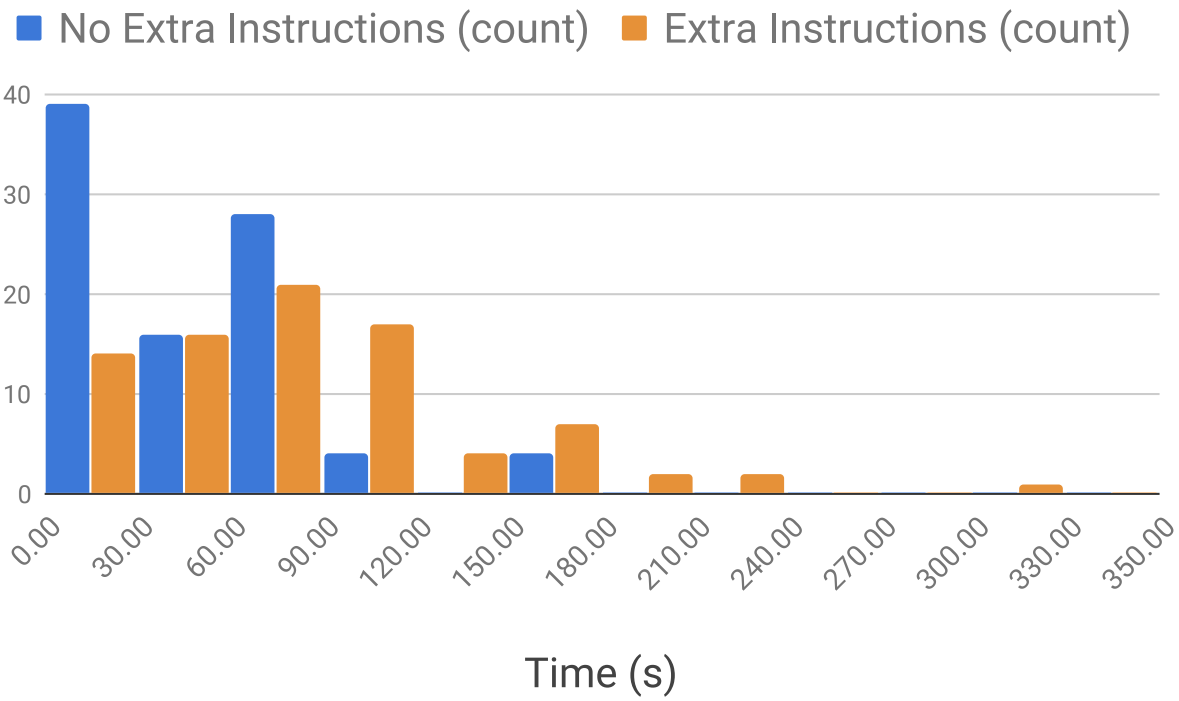


Figure 4.5: *Ballot Submission Times for Different Instructions.* Histogram showing the time from ballot printing to ballot submission for two sets of experiments: ones where participants were given instructions designed to increase verification and ones where participants received standard instructions. Participants in the former group took longer: 83 s on average vs. 50 s for those who received no special instructions. Voters that received extra instructions but who were not given a slate took an average of 62 s.

($N = 220$) detected errors with their ballots 19% of the time, while their those who did not ($N = 21$) detected errors 4.8% of the time. Those who reported voting previously ($N = 216$) caught ballot discrepancies in 19% of cases, again performing better than those who reported not voting before ($N = 25$), who detected an error in 4.0% of cases. If someone reported voting in the 2018 midterm election ($N = 209$), they detected problems with their ballot 20% of the time, whereas if they did not ($N = 32$), they detected problems 3.1% of the time. This may indicate that familiarity with the midterm ballot we used caused participants to feel more invested in the accuracy of their votes; however, we did not establish this to statistical significance.

Other demographic factors, such as age, education, ethnicity, and gender, had no correlation with detecting manipulation.

4.4.3.4 Ballot position

Noticing was correlated with ballot position (Pearson's of -0.64), indicating that discrepancies in more prominent races are more likely to be noticed. (Race 0 was the first race on the ballot, so the number of noticers decreases as the race position increases, hence the negative correlation coefficient.) On our ballot, the first five races (Governor, Secretary of State, Attorney General, U.S. Senator, and Representative in Congress) were prominent partisan contests with a high likelihood of name recognition. In the experiments with no intervention (**E1–E3**), 37 participants had one of these races manipulated, and five reported the error on the exit survey, a rate of 14%. Kortum et al. actually observed the opposite of this effect, although neither our results nor theirs could establish an effect with statistical significance [132]. Additional experiments are necessary to establish whether this effect exists.

4.4.3.5 Undervotes

A metric that may inform voters' ability and willingness to verify their ballot is how much care they take in filling out the ballot. There are two metrics we use to examine this: whether a participant voted in every contest on the ballot, and whether the participant voted in every available position on the ballot (e.g., in a vote-for-two contests, the participant selected two choices). Table 4.2 shows the rates of voting in every race and every position on the ballot, with **E8** and **E9** removed as they directed participants to vote in every position. Voters who noticed discrepancies voted in every race or every position at a higher rate than those who did not, but not significantly so (likely due to our small sample size). Since these undervotes are visible to malware running on a BMD, this correlation could be exploited by an attacker to focus cheating on voters who are less likely to carefully verify, provided future work more firmly establishes this link.

	Overall	Noticers	Non-noticers
Every race	64.3%	73.9%	63.0%
Every position	43.0%	47.8%	42.4%

Table 4.2: *Participant Attentiveness.* Voters who noticed the discrepancy tended to vote in every race and ballot position more often than those who did not.

4.4.3.6 Partisanship

To assess the role partisanship plays in detection rates, we scored each ballot with a partisanship score, where a vote for a Democratic candidate was scored -1 and a vote for a Republican candidate was scored 1, and we take the absolute value of the sum. There were 11 opportunities to vote in a partisan way, so a participant who voted straight-party for either major party would achieve a score of 11. Excluding **E8** and **E9**, where voters were directed how to vote, the mean partisanship score for our participants was 8.3, and the median was 11. Although our BMD did not offer an automatic "straight-party" voting option, 105 participants achieved the maximum partisanship score.

Intuitively, a voter expecting every selected candidate to be from the same party might be more likely to notice a selection from a different party. Looking at only these straight-party voters, 15 out of 105 detected the errors. Of those, nine had a partisan race swapped to a different candidate of a different party, and six of those participants wrote in the survey that they had detected the change based on party. For example, one participant wrote, *"voted GOP for governor / lieutenant governor but Libertarian was actually selected on the paper ballot."*

This suggests that choosing a uniform set of candidates may help voters detect when something has gone wrong on their ballot, although more work is needed to establish that this is indeed the case, especially in more politically diverse populations. If this positive effect holds, it could be further promoted with ballot designs that prominently display the party, which could help voters see the information that is important to them while they review the ballot. On the other hand, BMD malware could be designed to counter this effect by focusing cheating on voters who do not cast a straight-party ballot.

4.4.3.7 Slate voting

34 participants were assigned an intervention which asked them to vote for a preselected slate of candidates (with a partisanship score of 0). Of these, only 26 participants voted exactly as directed. Of the eight participants who did not, four voted a straight Democratic ticket (partisanship score of 11), one voted a heavily Democratic ticket (score of 9), two voted slightly Democratic tickets

(scores of 3 and 5), and one voted a non-partisan ticket (score of 0), which only deviated from the slate in five positions. Of the eight participants who deviated from the slate, no participant deviated by fewer than five positions, indicating that either the deviation was deliberate or our instructions to vote the slate were unclear. Only one deviating participant managed to notice the discrepancy on their ballot, leaving participants who deviated from the slate a 13% notice rate compared to the 73% notice rate for those who did not deviate.

4.4.3.8 Network effects

One potential feature of a live polling place environment is a network effect: will a voter who is voting at the same time as a noticer be more likely to notice a problem on theirs? However, the number of people who notice in a given experiment is a confounding factor: voters are more likely to overlap with a noticer if there are more noticers. To interrogate this, we ran partial hypothesis tests for each intervention using Fisher's exact tests with permutations of overlapping with a noticer and noticing, and then combined using Fisher's combining function. We found that the effect of overlapping with a noticer did not significantly impact whether a participant noticed. This suggests that our interventions were more important than overlapping.

4.4.3.9 Signage

One feature that did not correlate with improved verification performance was the signage we tested (**E4**). Our observer noted that 11 of 30 participants in the signage experiment did not notice the sign at all. Only two participants in this experiment detected the modification of their ballot and reported it, and only one accurately noted the discrepancy in their survey, suggesting that passive signage alone may be insufficient to capture voters' attention and shape their subsequent behavior.

4.4.4 Participant Comments

Participants had two free-response sections in the exit survey. The first asked about anything "odd" they had noticed about the ballot. The second invited any additional comments. Of the 241 participants, 114 responded to at least one of these prompts. We note several features of their responses.

4.4.4.1 Discrepancy reports

In total, 44 participants (18%) noted in the free response section of the survey that they had identified some discrepancy on their paper ballot. Of these, 31 correctly identified the change, 12 gave no detail (e.g., *"At least one of my choices did not match who I picked"*), and one incorrectly identified the change (but did report that there was a mistake). We omitted this last participant from our "noticers" category where applicable.

Of the 44 participants who reported a change on their ballot in the survey, five added that they thought it could have resulted from a mistake they made. For example, one participant reported: *"I don't remember voting for the member of Congress and there was a vote. I very well may have but just don't remember."*

4.4.4.2 Attitudes about verification

Twelve participants mentioned either that they would only be comfortable voting on a paper ballot or that they were comforted by the fact that a paper trail was created. Only three of these 12 participants noticed that their ballot had been modified, despite the fact that they recognized that the paper ballot was an important tool for ensuring election integrity.

Several participants seemed to realize *after* casting their vote that the evaluation of their paper ballot was important; 13 participants mentioned in the survey that they did not review or that they should have reviewed the ballot, although we did not ask them about it. This concern may have been triggered by our survey question about what they had noticed about the paper ballot, but it also might be an indication that our interventions did cause voters to think about the risk—albeit too late. This would comport with Bravo-Lillo et al.'s findings about the ways novice users respond to warnings: after the fact [51].

The free responses also indicate that some participants assumed that the vote was completed and submitted on the BMD, rather than the paper ballot being the official record of their vote. One participant wrote, *"I was surprised to still have a paper ballot, after using the touch system. I was expecting the results to be registered electronically."* This assumption may discourage voters from verifying the selections on their paper ballot. Similarly, another participant, prompted by script variant 3 (“Have you carefully reviewed each selection on your printed ballot?”), responded to a poll worker, *“I checked it on the screen, it better be right.”*

Three participants expressed concern that they would not know what to do if they noticed a problem with their paper ballot during a real election. One person wrote, *“Having the printout be incorrect was confusing and it’s not clear how that would be handled in an election environment.”*

4.4.4.3 Feedback on the BMDs

We told participants that the experiment was a study about a new kind of voting system, and many left feedback about the interface and appearance of the machines. In Michigan, where we conducted the study, BMDs are available in every precinct, but voters must request to use them. The vast majority of voters use hand-marked paper ballots, so study participants were likely unfamiliar with BMD voting. In their comments, 21 participants expressed liking the system, while only three disliked it. Although merely anecdotal, this reflects previous findings that voters like touch-screen voting equipment [96].

4.5 Security Model

We are primarily motivated by the threat of undetected changes to election outcomes due to BMD misprinting attacks. Prior work has shown that such attacks cannot be reliably ruled out by pre-election or parallel testing [218], and we seek to answer whether voter verification can be an effective defense.

If a voter reports that their printed ballot does not reflect their on-screen selections, what should election officials do? Unfortunately, there is not yet a practical way to prove that the BMD misbehaved during voting. From officials’ perspective, it is possible that the voter is mistaken, or even lying, and in a large voter population, there will always be some rate of spurious problem reports, even when BMDs are working correctly. In principle, voters can vote on the BMD in front of an official to demonstrate the error and provide evidence that the machine is misbehaving. This does not necessarily violate the voter’s secret ballot, so long as the ballot they vote to demonstrate the problem is not cast. However, it seems unlikely that this will appear to voters as if they are not disclosing for whom they wish to vote and thus may be of little comfort.

For these reasons, problem reports from voters can serve only as evidence that something *might* be wrong with the BMDs. If the evidence exceeds some threshold, officials could invoke contingency plans. For instance, they could remove reportedly misbehaving BMDs from service to minimize further damage, perform forensic investigations in an attempt to uncover the cause, or even rerun the election if outcome-changing fraud cannot be ruled out.

Any of these responses would be costly (and none is foolproof), so the threshold for triggering them should not be too low. Moreover, attackers could exploit a low threshold by recruiting voters to fraudulently report problems, in order to disrupt or discredit the election. On the other hand, if the threshold is too high, outcome-changing fraud could be ignored.

To better understand how verification performance affects security in this setting, we construct a simple model. We assume, optimistically, that the attacker has no way to guess whether a particular voter is more likely than average to detect the alteration, and so chooses voters to attack at random. Based on our findings, this assumption seems reasonable given the available evidence. We further assume that whenever voters detect problems, they are able to remedy them and cast a correct vote by hand-marking a ballot. Except where noted, the model assumes that all voters cast their votes using BMDs.

Number of problem reports Let d be the fraction of misprinted ballots that voters detect, report, and correct. Suppose a contest had n ballots cast, and the reported fractional margin of victory was m. To have changed the outcome, the attacker would have had to successfully modify at least $n\frac{m}{2}$ cast ballots. However, since some modifications would have been corrected, the attacker would have had to induce errors in a greater number of printouts: $n\frac{m}{2(1-d)}$. Under our optimistic assumptions, if the attack changed the outcome, we would expect the fraction of voters who reported problems, a, to exceed:

$$a > m\frac{d}{2(1-d)}.$$

The model shows that the security impact of verification is non-linear, because every voter who corrects an error *both* increases the evidence that there is a problem *and* forces the attacker to cheat more in order to overcome the margin of victory. Figure 4.6 illustrates this effect.

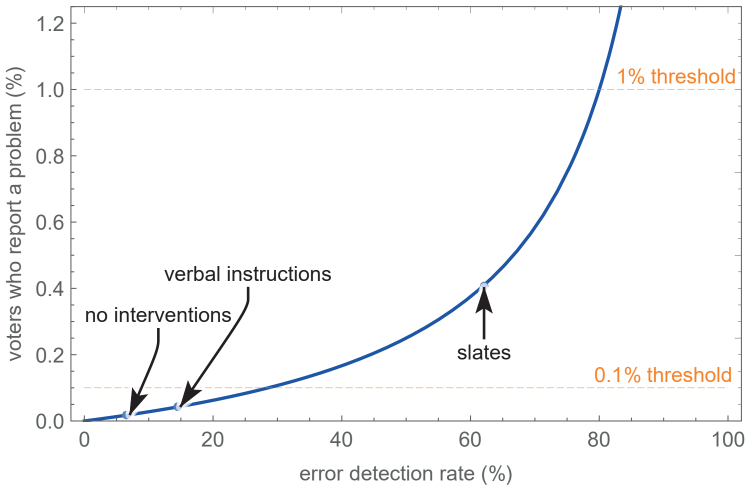


Figure 4.6: *BMD security is highly sensitive to human performance.* Given a 0.5% margin of victory, we plot the percentage of voters who report a problem during the minimal outcome-changing attack as a function of the rate at which errors are detected and corrected. This model implies that using BMDs safely for all voters requires dramatically improved verification performance or very sensitive attack detection thresholds.

With the 6.6% error detection rate from our non-intervention experiments and a close election with a 0.5% margin (the margin that causes an automatic recount in many states) a successful attack would cause as few as 0.018% of voters—less than 1 in 5000—to report a problem. Small changes in verification performance around our base rate cause relatively little change in the amount of evidence. More than doubling the error detection rate to 14% (the rate we found for prominent races) only increases the fraction of voters who report a problem to 0.039%. However, larger improvements have an outsized effect: with the 86% error detection rate from our most successful experiment, at least 1.5% of voters (1 in 67) would report problems.

Required detection rate Suppose election officials activate a countermeasure if the fraction of voters who report problems exceeds a threshold a^*. For a given margin, the countermeasure will be triggered by minimal outcome-changing fraud when:

$$d > \frac{2a^*}{m + 2a^*}.$$

An expensive countermeasure, like rerunning an election, will require a high trigger threshold—say, 1% of voters reporting a problem—to avoid false positives. With a 0.5% margin, reaching a 1% trigger threshold would require an error detection rate exceeding 80%. A less expensive countermeasure, such as an investigation, might be triggered by a lower threshold—say, 0.1%. Reaching this lower threshold in an election with a 0.5% margin would require an error detection rate greater than 29%. This suggests that using BMDs securely for all voters will require large improvements to verification performance or extremely low thresholds for triggering countermeasures.

Minimizing BMD voting helps dramatically Securing against misprinting attacks is far easier if only a small fraction of voters use BMDs than if all in-person voters do. This is because an attacker would be forced to cheat on a much larger fraction of BMD ballots in order to achieve the same change to the election results. Moreover, if the population of BMD voters is smaller than half the margin of victory, it is impossible for a BMD misprinting attack to change the outcome.

Let b be the fraction of voters who use BMDs. We can replace m in the expression above with $\frac{m}{b}$ and let a^* be the fraction of *BMD voters* that must report a problem to trigger the countermeasure. In Maryland, which uses hand-marked paper ballots but makes BMDs available to voters who request them, 1.8% of voters use BMDs [149]. With a 0.5% margin, as in the previous example, Maryland would reach a complaint threshold of 1% of BMD voters with an error detection rate of only 6.7%. If 5% of voters use BMDs, the error detection rate would need to be 17%.[4] Our results suggest that these more modest rates of verification likely are achievable, in contrast to the far greater accuracy required when all voters use BMDs.

This model overestimates security An attacker might use any number of features (including

[4] Maryland has since updated their policy to offer both hand-marked and BMD ballots to every voter [102]. Approximately 10% of voters now vote on BMDs. For the same setup, approximately 30% of BMD voters would need to detect and correct problems, which is attainable given our experimental results.

several of the correlations we observed) to focus cheating on voters who are less likely to successfully catch errors. For instance, an attacker could preferentially modify ballots that have undervotes or a mix of selections from different parties. Attackers could also selectively target voters with visual impairments, such as those who use large text or an audio ballot. Other features, such as how long voters spend inspecting the candidate review screen, might also prove to be predictive of verification success. For these reasons, our simplified model is likely to overestimate the effectiveness of verification against sophisticated attackers.

We also note that some attackers may merely seek to cast doubt on election results by causing highly visible errors or failures—which are also possible with hand-marked paper ballots. However, in general, BMDs are vulnerable to all classes of computer-based attacks that affect hand-marked paper ballots and to others, such as the misprinting attack discussed here, to which hand-marked paper ballots are not susceptible.

4.6 Discussion

4.6.1 Limitations

It is challenging to capture real-world voter behavior in a mock election. However, our study followed established best practices [174], and we strived to create as realistic a polling environment as we could. It is impossible to know exactly how well we succeeded, but the effect seems to have been convincing: several people approached us to ask whether there was a real election taking place that they had not heard about. Our participants also seemed engaged in the study; many expressed strongly held political preferences in our survey (so much so that some refused to vote according to our slate), and a large majority reported voting in the 2018 midterm. On the other hand, the election used a ballot that was more than nine months old, which may have reduced participant motivation, and we had a few participants who reported that they did not vote in our state or were otherwise unfamiliar with our ballot. It is also possible that our results were skewed due to selection bias and observer effect.

Another limitation of our work is that we drew participants from a population that is locally but not nationally representative. Our participants tended to be younger, significantly better educated, more liberal, more likely to be female, and more likely to be Caucasian than the average voter in the United States [229]. Future work is needed to validate our study in more diverse and representative populations.

Although our results suggest that certain interventions can boost verification performance, the data is too sparse to provide a high-fidelity understanding of the magnitude of the improvements. In addition, due to time constraints, we were unable to test the interplay of all combinations of interventions, and some interventions appear to be sensitive to small changes (e.g., the difference in phrasing between script variants 2 and 3). Further study is needed to better characterize what makes interventions work and how they interact before we can confidently conclude that any particular set of procedures will be effective in practice. It is worth noting that most of our results have been replicated in a more recent work [132], which is encouraging.

4.6.2 Discussion of Findings

Our study provides the first concrete measurements of voter error detection performance using BMDs in a realistic voting environment. At a high level, we found that success rates without intervention are very low, around 6.6%. Some interventions that we tested did not significantly impact detection rates among participants, although others improved detection drastically and may serve as a roadmap for interventions to explore in further research. We discuss those interventions here.

4.6.2.1 Verbal instructions can improve verification

Notably, all interventions that involved poll workers verbally encouraging verification between the BMD and the scanner—those in **E6–E9**—resulted in higher ballot reviewing and error reporting rates. This, coupled with the fact that reviewing the printout was highly correlated with error detection across all of our results, suggests that interventions focused on causing the voter to review the ballot carefully may be helpful. On the other hand, instructions at the beginning of the voting process (**E5**) and passive signage (**E4**) had no significant effect on error reporting. This pattern of effects is supported by findings from the usable security literature, which suggest that post-completion errors can be mitigated with timely interruptions that encourage individuals to take defensive steps [55]. It is also some comfort to notions of evidence-based elections [214], software independence [194], and cast-as-intended to know that while the existence of evidence is not enough, there are simple policy interventions that make usage of the evidence robust.

It is worth noting that we also found that these interventions caused participants to take longer to submit their ballots, on average about twice as long. This could cause longer lines at polling places if these interventions are implemented without complementary procedural considerations, such as having adequate space for voters to stop and review their ballots.

4.6.2.2 Effectiveness of slates

Directing participants to vote for a provided slate of candidates, combined with verbally prompting them to review their printouts, resulted in strongly increased rate of error detection: 74% of participants who were given a slate and did not deviate from it noticed the errors. This finding may suggest that encouraging voters to write down their preferences in advance can boost verification.

However, the slates we used functioned quite differently from slates likely to be used in practice. The choices we provided were randomly generated and had no basis in the subject's preferences—in a real election, slates would reflect for whom the voter intended to vote, most likely created by the voter or their political party [123]. It is possible that the success rate we observed was primarily due to participants carefully attempting to follow our instructions and vote for unfamiliar candidates. Further study is needed with more realistic slate conditions (i.e., asking subjects to write down their preferences) in order to assess whether slates really do help voters catch errors. We note that this potential flaw also exists in more recent work that relied on prescribed slates [132], however it is a best practice for the research setting [174].

4.6.3 Recommendations

Since BMDs are widely used today, we recommend several strategies for improving voter verification performance. While we are unable to conclude that these strategies will enhance error detection to the point that BMDs can be used safely in close or small elections, our findings indicate that they can help. These interventions are designed to be lightweight and relatively easy to drop into existing election policies and practices, and they have been implemented in jurisdiction across the United States, including the state of North Carolina [25]. They support my overall thesis that secure election technologies only work if they accommodate the real world.

4.6.3.1 Design polling places for verification

Polling place layout and procedures should be designed with verification in mind. As we have discussed, voters need time and space to verify their ballots. If tables or areas to stand out of the way are provided, voters will be able to carefully verify without causing lines to form or slowing polling place throughput. The presence of such a "verification station" might also encourage verification.

Another practical concern is privacy. Several of our participants expressed discomfort with the fact that we did not provide a privacy sleeve for their ballots (a requirement in Michigan), and that the scanner accepted the ballots face-up only, with one participant stating, *"I feel like inserting the ballot face up in the scanning machine will make people uncomfortable."* Voters may not feel comfortable reviewing their ballots in front of poll workers but may be unsure where to go to review them privately.

4.6.3.2 Incorporate post-voting verbal instructions

As all of our script-based interventions that took place after the ballot was printed (**E6–E9**) showed an increase in verification performance, we recommend that poll workers interrupt voters after their ballot has printed but before it is scanned and ask them to review it. Signage with a similar message to our scripts placed at the optical scanner (**E4**) or instructions before the participants voted (**E5**) did not result in significant differences in error detection; nevertheless, further study with additional variations is prudent before ruling out such strategies.

4.6.3.3 Encourage personalized slate voting

Although our study tested randomized slates, rather than personalized slates, the effect size was so large that we tentatively recommend encouraging the use of personalized slates by voters. In our experiments (**E8** and **E9**), participants who were directed to vote using a randomized slate (and did not deviate) reported errors at a rate of 73%. If voters prepare their own slates at home (or use a printed slate prepared, for instance, by a political party or other organization), they can use them to check each selection on the BMD printout. We note that, since we did not directly test the use of personalized slates, further research is necessary to ascertain whether large performance gains are actually achieved. Furthermore, even if personalized slates are effective, the gain will be limited to the fraction of voters who can be induced to use them.

Slates have potential downsides and should be used with care. They have the potential to compromise ballot secrecy, so we recommend providing a closed trash can, paper shredder, or other means for voters to privately dispose of them before leaving the precinct. Coercion is also a threat, but voters could be advised to prepare multiple different slates as a defense.

4.6.3.4 Help voters correct errors, and carefully track problems

Verification-promoting interventions will be of little use if action cannot be taken to remedy misbehaving BMDs—something that even our participants expressed concern about.

First, it is crucial that polling places have a procedure for voters who want to correct their printed ballots. Several subjects commented that they would not know what to do if something was wrong with their ballot in a real election, indicating that this problem is present in current election procedures.

Second, detailed records should be kept about which BMD the voter used and what the specific issue was, including the contest and candidates involved (to the extent that the voter is willing to waive ballot secrecy). Problems should be treated as potentially serious even when the voter believes they are at fault—we note that several participants in our study believed they had made a mistake even though the BMD actually was programmed to be malicious. Problem reports should be centrally reported and tracked during the election, so that issues affecting multiple precincts can be identified as rapidly as possible.

4.6.3.5 Prepare contingency plans

What to do in the event that BMDs are known or suspected to be misbehaving is a more difficult question. If an elevated number of voters have a problem with a single machine, it should be taken out of service, provided there are other BMDs available for use (especially for voters with disabilities, who may have no alternative).

If widespread problem reports occur—particularly problems focused on a tightly contested race or significantly exceeding the rate reported in past elections—officials could consider taking most BMDs out of service and encouraging all remaining voters who can to use hand-marked ballots. This raises logistical challenges: polling place would need to have enough ballots available for hand-marking, or the ability to print ballots on demand, and votes already cast on the BMDs would be suspect.

After the election, forensic analysis of the BMDs could be performed to attempt to determine the cause of reported errors. Unfortunately, such analysis cannot in general rule out that a sophisticated attack occurred and left no digital traces. Even if programming errors or attacks are uncovered, they may be impossible to correct if officials are unable to determine whether the effects were large enough to change the election outcome. The only recourse might be to re-run the election.

Our findings show that, in the event of an actual error or attack, the rate of reported problems is likely to be only the tip of the iceberg. In our non-intervention experiments, undetected errors outnumbered reported problems by almost twenty to one. Our results further suggest that an attacker who cleverly focused cheating on voters who were less likely to verify could achieve an even higher ratio of undetected errors. An effective response requires either being very sensitive to reported problems—which increases the chances that an attacker could trigger false alarms—or achieving very high error correction rates.

4.6.3.6 Educate voters about BMD operations and risks

Like in other human-in-the-loop security contexts, greater education could boost voters' awareness of the importance of careful verification and boost error detection and reporting rates.

To this end, we recommend educating voters that the paper, rather than what the BMD screen shows, is the official record of their votes. Several of our participants said they realized after scanning that they should have, but did not, review their printouts. Others stated that they had checked the review screen on the machine and that they trusted the paper to be correct. It is likely that many participants incorrectly assumed that the BMDs, rather than the paper and scanner, tabulated their votes.

We also recommend educating voters about the possibility of BMD malfunction. Many of our participants seem not to have even considered that the machine might have changed their votes, as indicated by the voters who blamed themselves for the misprinted ballots. Raising threat awareness could help motivate voters to carefully inspect the paper, as well as give them greater confidence to report any discrepancies they detect. Jurisdictions like Johnson County, Kansas appear to have taken these recommendations to heart [122] with a new social media campaign.

4.6.3.7 Consider the needs of voters with disabilities

Further research is needed to specifically examine verification performance among voters with disabilities, but we offer some initial recommendations here. Detecting errors in printed ballots may be especially challenging for voters with impaired vision. Designing BMD ballots for maximum legibility might help, and so might encouraging voters who use text-to-speech devices to bring them to the polls for use during verification. Jurisdictions could also provide air-gapped accessible devices to read the ballot back to voters, in case voters do not have their own text-to-speech devices. These steps would have the added benefit of reinforcing the message that the content of the paper ballots is what gets counted. If BMDs are to live up to the promise of better and more accessible voting, enabling all voters to verify their printed ballots is a must.

4.6.3.8 Require risk-limiting audits

Even perfectly verified paper ballots are of little use for security if they are not rigorously audited to confirm the results of computer-based tabulation. Fortunately, risk-limiting audits [143] (RLAs) are gaining momentum in the United States. Colorado, Nevada, and Rhode Island mandate statewide RLAs, and states including Michigan, Virginia, Ohio, Georgia, and Pennsylvania have begun pilots [80]. RLAs and effective verification are both necessary in order for paper to provide a strong defense against vote-stealing attacks, and we recommend that efforts to achieve both be pursued vigorously. More discussion of RLAs is provided in the next two chapters.

4.7 Conclusion

This chapter presented results from the first empirical study of how well voters using BMDs detect errors on their printed ballots, which is a limiting factor to the level of security that a BMD-based paper trail can provide. Based on the performance of 241 human subjects in a realistic polling place environment, we find that, absent specific interventions, error detection and reporting rates are dangerously low. Unless verification performance can be improved dramatically, BMD paper trails, particularly when used by all in-person voters, cannot be relied on to reflect voter intent if the machines are controlled by an attacker. This represents a failure of an election security technology, voter-verified paper, in the face of real world challenges.

Nevertheless, we also find that procedural interventions can improve rates of error detection and reporting, potentially increasing the security offered by BMDs. The interventions we tested should serve as examples of what is and is not likely to be effective, and we hope they will point the way for further research and experimentation. These findings add to the broad literature of human-in-the-loop security results and recommendations, and they provide additional examples of what does and does not work in human-centric security.

Our results should not be read as demonstrating that BMDs can be used securely. Further work is needed to explore the potential for attackers to predict which voters will verify, and additional human-subjects testing is necessary to confirm whether sufficient rates of verification success can be achieved in practice. The cost of implementing interventions and contingency plans may also be prohibitive. Nevertheless, BMDs do offer advantages, including uniform accessibility and ease of administration. We hope our work will help election officials make better informed choices as they weigh these benefits against the security risks of using BMDs for all voters.

CHAPTER V

Modeling Post-election Audits

5.1 Introduction[1]

As we have observed so far, no method for counting votes is perfect, and methods that rely on computers are particularly fragile: errors, bugs, and deliberate attacks can alter results. The vulnerability of electronic voting was confirmed in two major state-funded studies, California's Top-to-Bottom Review [48] and Ohio's EVEREST study [155]. More recently, at the 2017 and 2018 DEFCON hacking conferences, attendees with little or no knowledge of election systems were able to penetrate a wide range of U.S. voting machines [40, 41]. Given that Russia interfered with the 2016 U.S. Presidential election through an "unprecedented coordinated cyber campaign against state election infrastructure" [232], national security demands we protect our elections from nation states and other advanced persistent threats.

There is no way to secure a software system perfectly, which is why we need *software-independent voting systems* [194]. A voting system is software-independent if an undetected change or error in its software cannot cause an undetectable change or error in an election outcome. The most practical way to achieve software independence is by using voter-verifiable paper records (as just discussed, a non-trivial requirement), and manually auditing them to check whether reported outcomes based on electronic tallying are correct. Risk-limiting audits give a criterion for the mini-

[1] This section has is based on "Bernoulli Ballot-Polling: A Manifest Improvement for Risk-Limiting Audits" in conjunction with Kellie Ottoboni, J. Alex Halderman, Ronald L. Rivest, and Philip B. Stark that appeared in *Proceedings of the 4th Annual Workshop on Advances in Secure Electronic Voting*

mal amount of scrutiny the ballots must receive: at least enough to establish, with high confidence, that the reported winner or winners of a contest really won.

The *outcome* of an election contest is the winning position(s) or candidate(s), not the numeric vote totals. An election outcome is *correct* if it is the outcome that would be found by a correct application of the social choice function to voter intent manually ascertained from the voter-verifiable marks on all ballots validly cast in the election.

A *risk-limiting audit* (RLA) of an election contest using a trustworthy record of voter intent[2] is a procedure with two possible endpoints:

1. Declare that the election outcome (i.e., the set of winners) is correct.

2. Declare that there should be a full manual tally of the record to determine the correct outcome.

The *risk limit* of an RLA is the chance that the audit ends by declaring the outcome to be correct, when in fact the election outcome is incorrect for any reason (including human error, software bugs, and malicious hacking).

Risk-limiting audits (RLAs) were introduced in 2007 [215] as a mechanism for detecting and correcting outcome-changing errors in vote tabulation, whatever their cause—including hacking, misconfiguration, and human error. RLAs have been tested in practice in California, Colorado, Indiana, Virginia, Ohio, Michigan, Pennsylvania, Missouri, and Denmark. Colorado started conducting routine statewide RLAs in 2017 [141], and Rhode Island passed a law in 2017 requiring routine statewide RLAs starting in 2020 (RI Gen L § 17–19–37.4), as has Nevada. RLA legislation is under consideration in a number of other states, and bills to require RLAs have been introduced in Congress.

The American Statistical Association has endorsed risk-limiting audits as best practice, as have the Presidential Commission on Election Administration, the League of Women Voters, the Verified Voting Foundation, Common Cause, and other election integrity organizations [140].

[2] The trustworthiness of the audit trail needs to be established by a *compliance audit*. See, e.g., [30, 214].

However, many lessons have been learned in the practical implementation of RLAs, and it has become apparent that certain features like the total number of ballots likely to be hand-tabulated are not as important to election officials considering adopting RLAs as previously thought. Empirical data gathered from pilots in Rhode Island [108] and Michigan [1] has shown that while audits in the abstract do not take much time, other factors, like a desire to finish in one round, are most important. Questions of financial cost are often cited as important decision factors, and to date there is not a robust way to provide cost estimates for any kind of post-election auditing technique that is not a fixed audit. Comparisons between RLA techniques are also difficult to make for policy makers, as doing so requires wading into complex statistics to provide apples-to-apples comparison.

In this chapter I present a workload estimation function that can be used to directly compare various post-election auditing methods. This function is derived from empirical data collected in Rhode Island during pilots of RLAs, and validated against subsequent RLAs performed in Rhode Island, Michigan, and Colorado. This workload estimate enables an apples-to-apples comparison of various RLA techniques, revealing some previously-unknown details, for example that ballot-polling RLAs become less efficient than batch comparison RLAs for close margins. The estimator also reveals and quantifies something that was previously known but not rigorously examined: audits are highly parallelizable.

The rest of this chapter is laid out as follows: in the next section I provide more detailed background on RLAs than I have up to now. In Section 5.3 I develop my workload estimation function, and in Section 5.4 I discuss how post-election audits are highly parallelizable. In Section 5.5 I validate my parallelized workload model against real-world data. In Section 5.6 I use the function to compare three popular RLA techniques, and I discuss the limitations of this comparison in Section 5.7. I conclude in Section 5.8 and setup a discussion for an improved form of ballot-polling RLAs based on these observations about parallelizability, which I present in the next chapter.

5.2 What is a risk-limiting audit?[3]

There are two general approaches to risk-limiting audits:

- *comparison audits*, which compare the machine interpretation to a manual interpretation of randomly selected ballots or batches of ballots, and

- *ballot-polling audits*, which rely only on a manual interpretation.

Both methods rely on the existence of a *ballot manifest* that describes how the audit trail is stored. Selecting the random sample can include a public ceremony in which observers contribute by rolling dice to seed a PRNG [79].

Ballot-polling audits examine random samples of individual ballots. They demand almost nothing of the voting technology other than the reported outcome. When the reported outcome is correct, the expected number of ballots a ballot-polling audit inspects is approximately quadratic in the reciprocal of the (true) margin of victory, resulting in large expected sample sizes for small margins.

Comparison audits compare reported results for randomly selected subsets of ballots to manual tallies of those ballots. Comparison audits require the voting system to commit to tallies of subsets of ballots (“clusters”) corresponding to identifiable physical subsets of the audit trail. Comparison audits have two parts: confirm that the outcome computed from the commitment matches the reported outcome, and check the accuracy of randomly selected clusters by manually inspecting the corresponding subsets of the audit trail. When the reported cluster tallies are correct, the number of clusters a comparison audit inspects is approximately linear in the reciprocal of the reported margin. The efficiency of comparison audits also depends approximately linearly on the size of the clusters. Efficiency is highest for clusters consisting of individual ballots: individual cast vote records. To

[3] Text in this section was borrowed from “Public Evidence from Secret Ballots”, work in conjunction with Josh Benaloh, J. Alex Halderman, Ronald L. Rivest, Peter Y. A. Ryan, Philip B. Stark, Vanessa Teague, Poorvi L. Vora, and Dan S. Wallach that appeared in *Proceedings of the 2nd International Joint Conference on Electronic Voting* [35] and “Bernoulli Ballot-Polling: A Manifest Improvement for Risk-Limiting Audits” in conjunction with Kellie Ottoboni, J. Alex Halderman, Ronald L. Rivest, and Philip B. Stark that appeared in *Proceedings of the 4th Annual Workshop on Advances in Secure Electronic Voting* [175], as well as some new text.

audit at the level of individual ballots requires the voting system to commit to the interpretation of each ballot in a way that is linked to the corresponding element of the audit trail.

In addition to RLAs, auditing methods have been proposed with Bayesian [191] or heuristic [196] justifications.

All post-election audits implicitly assume that the audit trail is adequately complete and accurate that a full manual count would reflect the correct contest outcome. *Compliance audits* are designed to determine whether there is convincing evidence that the audit trail was curated well, by checking ballot accounting, registration records, pollbooks, election procedures, physical security of the audit trail, chain of custody logs, and so on. *Evidence-based elections* [214] combine compliance audits and risk-limiting audits to determine whether the audit trail is adequately accurate, and if so, whether the reported outcome is correct. If there is not convincing evidence that the audit trail is adequately accurate and complete, there cannot be convincing evidence that the outcome is correct. As we saw in the last chapter, the paper trail must also be verifiably correct as a representation of the voters' selections; if the paper trail does not reflect the will of the voters, no amount of auditing can prevent fraud.

5.2.1 Audits in Complex Elections

Generally, in traditional and complex elections, whenever an election margin is known and the infrastructure for a comparison audit is available, it is possible to conduct a rigorous risk-limiting comparison audit. This motivates many works on practical margin computation for IRV [45, 63, 148, 202]. Recent breakthroughs have been made by Blom et al. [43, 44].

However, such an audit for a complex election may not be efficient, which motivates the extension of Stark's *sharper discrepancy measure* to D'Hondt and related schemes [220]. For Schulze and some related schemes, neither efficient margin computation nor any other form of RLA is known (see [115]); a Bayesian audit [68, 191] may nonetheless be used when one is able to specify suitable priors, as discussed by Vora et al. [159, 237]. Stark has also recently reframed RLAs as sets of half-averaged nulls, leading to still more efficient audits for simpler and complex elections [219]. Ottoboni et al. also developed RLAs for stratified samples, in cases where one type of audit could be performed over one set of ballots but not over the whole set [176].

5.2.2 Audit Units

RLAs work with one of two types of unit: batches and ballots. A batch is a collection of ballots corresponding to some level of results reporting. For example, all of the votes in one polling location may constitute a batch, or all the votes in a vote-tabulation district [53]. While a batch typically also corresponds to a physical container where its ballots are located, occasionally multiple batches can be stored in one container, or one batch can be stored in multiple containers. For the purposes of our discussion below, we will assume each batch gets its own container. Batch-level audits typically select a sample from the set of batches, and then every ballot in each of those batches is tabulated for the audit.

Ballot-level audits are similar to batch audits, except rather than counting every ballot in a batch, only certain ballots from certain batches will be selected. Typically, ballot-level audits are much more efficient than batch audits, as the number of ballots that they audit is much smaller, though as we shall see this is not always the case.

5.2.3 A general RLA

An RLA A has several inputs: the risk-limit, α, the reported outcome for the particular race R, x, the random seed used for generating random samples, the ballots B and batches P that are being audited, the test statistic T, and the hypothesis test H. We can write an audit as a function of these variables, $A(\alpha, R, x, B, P, T, H)$.

RLAs typically have a sample size estimation function E_A, that produces an expected number of samples needed to satisfy the RLA requirements. E_A can depend on the risk-limit, the reported outcome of the election, ballots, batches, test statistic, and prior sample S': $E_A(\alpha, R, B, P, T, S')$

RLAs also have a sampling function S_A, that can be defined in myriad ways. It depends on the random seed, sample size, as well as the batches and ballots, and returns a set of batches and ballots that should be audited, $S = \{P_A, B_A\}$. The sampling function is thus $S_A(x, B, P, |S|)$. Random sampling may rely on a variety of techniques, such as consistent sampling [193], geometric

Algorithm 1 RLA Procedure

```
procedure RISK-LIMITING AUDIT($\alpha$, R, B, P, T, H)
    $x \leftarrow RandomString$    ▷ Initialize $x$ with random data, usually from rolling dice
    $|S| \leftarrow E_A(\alpha, R, B, P, T, \emptyset)$    ▷ Figure out the initial sample size
    $S \leftarrow S_A(x, B, P, |S|)$    ▷ Draw the sample
    loop
        if $T_A(\alpha, R, B, P, T, H, S) < \alpha$ then
            return True    ▷ The audit has confirmed the outcome to risk-limit $\alpha$
        else if $|S| \geq |B|$ then    ▷ The audit has escalated to a full hand-recount
            if $Winner(R) \neq Winner(S)$ then    ▷ If sample has different winner than reported...
                return False    ▷ ... it found the winner didn't actually win
            else
                return True    ▷ The audit confirmed the outcome
            end if
        else    ▷ We don't have enough information yet
            $|S|' \leftarrow E_A(alpha, R, B, P, T, H, S)$    ▷ Compute a new sample size
            $S \leftarrow S_A(x, B, P, |S|')$    ▷ Draw the new sample, and repeat the test
            goto loop
        end if
    end loop
end procedure
```

skipping [176], or others [195].

Finally, RLAs have an testing function T_A that, given an audited sample of ballots, either confirms the outcome or requires the sampling of more ballots. T_A depends on the risk-limit, the reported outcome, the ballots, the batches, the test-statistic, the hypothesis test, as well as the sample generated by S_A.

A general form of an RLA is shown in Algorithm 1.

5.2.4 Mechanics of an RLA

In addition to the abstract steps required to complete an RLA, there is a fair degree of administrivia. To generate the random seed, typically twenty 10-sided dice are rolled by attendees to the RLA who are selected at random. Once the seed and contest information has been finalized, the RLA rounds can begin. At the start of each round, a sample is drawn based on a **ballot manifest**, a listing of all of the ballot containers and how many ballots are contained in each. To draw a sample, RLA software selects uniformly at random from a list of ballots constructed from the manifest. For example, if “Batch A” contains 150 ballots, then “Batch A, Ballot 1” through ‘Batch A, Ballot 150‘ would be appended to the list of ballots.[4]

Once the sample is drawn, the list of sampled ballots is typically broken up across teams of two or three people called **audit boards**. The audit boards are the people who locate and open ballot containers and count through the stack of ballots to find the selected ballots to audit. For the ballots that have been selected for audit, a placeholder is usually inserted in the ballot’s place in the stack, and the selected ballot is labeled and set aside.

Once all of the selected ballots have been pulled out of their batches, the audit boards inspect the marks on each one and “audit” the ballots. In a ballot-polling or a batch-comparison audit, this simply consists of tallying up the votes across all ballots in the sample, which audit boards can record on a paper spreadsheet or enter into the audit software directly. Votes for all choices are tallied, as well as undervotes and invalid votes.

For ballot comparison audits, once a ballot is found its corresponding **cast vote record** (CVR) is located. The CVR is the record of how the ballot was tabulated by the scanner on election day. The audit board directly compares the paper ballot in front of them with the CVR, recording whether there is a discrepancy between them. Discrepancies can include **one-vote misstatements**, where either the physical ballot shows a vote for the winning choice(s) but the CVR shows no vote (a one-vote understatement), or the CVR shows a vote for the winning choice(s) and the physical ballot does not (a one-vote overstatement). Additionally, if the physical ballot shows a vote for a losing choice but the CVR shows a vote for the winning choice, this is a two-vote overstatement (and a two-vote understatement for the inverse case).

Once all of the ballots in a round have been sampled, the sample results are compared with the reported results. For ballot-polling audits, this means comparing the totaled votes in the sample against the reported vote totals. For batch comparison, this means comparing the tallied results in each batch with the reported results for each sampled batch. For ballot comparison, this requires examining the total number of misstatements discovered by the audit. Once this data is collected,

[4] For RLAs that rely on cast-vote records, like ballot comparison audits, ballots may also have a unique identifier per-ballot to ease in CVR lookup.

the data is input to a hypothesis test, and if the resulting p-value is beneath the selected risk-limit, the audit can terminate. Otherwise, the audit will proceed to another round, until the risk-limit is met in one of the proceeding rounds or all of the ballots are audited.

5.3 Workload Estimation

An important first step in our examination of various RLA techniques is a robust definition of workload. One of the key features of RLAs is that they can provide high statistical confidence for a fraction of the work of a full hand recount. However, to date there is no universally applicable estimator of workload functions that applies across RLAs, so we define one below.

To construct such a function, we will rely on workload data from the Rhode Island RLA working group's report on RLA pilots [108]. To our knowledge, this is the only data set with robust timing measurements for the three major kinds of risk-limiting audit. However, the data reported in it do suffer some drawbacks. Sample sizes were small, so mean and median reporting times may not be accurate. Batches were small, with the largest batch containing only 324 ballots. However, despite these shortcomings, we believe that the overall results reflected in the data still provide useful information about the proportion of work involved in each step of a risk-limiting audit. We see later on the model we construct using this data does appear to be fairly true to reality based on experiences with other risk-limiting audits around the country. Further discussion of this is provided in Section 5.5.

5.3.1 Preliminaries

An RLA is performed over a set of ballots B contained in a set of containers P. It involves finding and opening containers containing paper ballots selected for audit, finding those ballots within the containers, and auditing them by recording the marks that appear on them. We assume both ballots and batches are well ordered according to the following rules, borrowing notation from [215]:

1. Ballots are in order $(b_{(i)})_{i=1}^{B}$ such that b_1 is the first ballot in a batch, b_2 is the second, and so on.

2. Batches are also in order, $(p_{(i)})_{i=1}^{P}$, so that p_1 is the first batch, and so on.

For an audit A over batches P and ballots B, we define the following functions:

$s(p)$, the time to locate a batch in storage:

$$s(p) \equiv 15s \tag{5.1}$$

While there is not much extant data on how long it takes to retrieve a batch on average, the Rhode Island RLA Working Group [108] report collected timing data for the time to retrieve a ballot from a new batch, including the time it takes to find the batch, open it, and find the ballot for a ballot comparison. The RIRLA report indicates this took $61s$ on average, and it took $31s$ to find a ballot in a batch that was already opened. Taking the difference, we obtain the time to find and open a batch at $30s$, which we split evenly between $s(p)$ and $o(p)$ (see below).

Note that using the RIRLA data in this way is imperfect, however, as we shall see later, tuning our model this way still leads to informative results. It is quite possible that in other audits, if batches are well ordered and stored, it may take less time to find any given batch. Similarly, if batches are not well stored, for example if some batches are in the warehouse across town, then finding batches may take more time.

$o(p)$, the cost of opening a batch:

$$o(p) \equiv 15s \tag{5.2}$$

Opening a batch for auditing requires some amount of work: cutting a seal, unzipping or unlocking the container, and doing some bookkeeping for chain of custody like writing down the seal number. As discussed above, while we do not have high fidelity data about how long this process takes in the course of an audit, we will again use the RIRLA report as a guide.

As before, this function may be significantly different depending on the specifics of an election, jurisdiction, and audit. If the batch is split up into multiple containers, as can happen if absentee ballots are sorted into their own container or if there are too many ballots to fit in one container, p_i

may correspond to multiple physical containers that need to be opened and their ballots combined.

$h(b)$**, the time of handling the physical paper that corresponds to ballot** b**:**

$$h(b) \equiv \begin{cases} 6s, & \text{in a batch comparison audit} \\ 31s, & \text{in a ballot comparison audit} \\ 77s, & \text{in a ballot polling audit with countdown} \\ 61s, & \text{in a ballot polling audit with } k\text{-cut} \\ 50s, & \text{in a ballot polling audit with ruler} \\ 10s, & \text{in a ballot polling audit with scales} \end{cases} \tag{5.3}$$

Certain audit types will require handling more ballots than others, as will certain methods of finding the ballot. The RIRLA report studied the three major categories of RLA, batch comparison, ballot polling, and ballot comparison. The reported recorded both handling and interpretation times for both methods. They found that organizing the ballots for interpretation and counting in a batch audit was much more efficient than in a ballot-level audit.

For ballot polling, the median time to retrieve and evaluate a ballot from an already open container using the countdown method was 104s, and the average time to evaluate and count one contest on a ballot that had already been pulled was 25s. Taking the difference, we obtain a time of 77s per ballot per contest using the countdown method. Of note, this time may be a bit inflated due to training difficulties and sample size (only eight ballots were audited using countdown). RIRLA also evaluated three other methods of sampling ballots in ballot polling: scales, rulers, and k-cut [212]. Times reported above were similarly calculated by taking the differences of the retrieval and evaluation time and the mean retrieval time. Again, this data is imperfect, since evaluation times were pooled across methods, but once ballots have been drawn, there should be relatively few differences in interpretation time.

Finally, the ballot comparison method took on average 31s to find a ballot in an already opened container.[5]

$t(b)$, the time to audit a ballot b

$$t(b) \equiv \begin{cases} 7s, & \text{in a batch comparison audit} \\ 25s, & \text{in a ballot polling audit} \\ 25s, & \text{in a ballot comparison audit} \end{cases} \tag{5.4}$$

For audited ballots, there is some cost to recording what is on them for the audit. For non-audited ballots, there isn't. The cost may be different depending on the audit method, the number of pages each ballot consists of, the number of races being audited, the number of candidates in the race, or the social choice function of the given election (e.g. methods like ranked-choice voting may take more time to record information for).

The Rhode Island report showed a $7s$ time to audit a ballot (that is, to interpret and record the marks that appear on it) in a batch comparison audit. The other two audit types took on average $25s$.

The assumptions baked into these preliminary functions are not hard and fast, and can be tailored to a particular jurisdiction's ballot storage procedures, chain of custody regulations, election specifics, and so on. However, for simplicity, we will use the values defined above in our analyses that follow as they are some of the only available empirical data.

5.3.2 The Workload Function

Now that we have captured costs for the discrete steps of sampling and auditing the ballots in an RLA, we can define a function W that captures the overall workload that a given RLA method entails. We will assume that all steps are taken linearly, without parallelization (see Section 5.4), so that each ballot in the sample is drawn one after another in sequence. While this is not how RLAs function in the real world (typically there are multiple audit boards that pull a subset of the sample at the same time), we choose it here for simplicity and for ease of comparing audit methods.

[5] The RIRLA report also noted that their batches were contained in folders within containers, and reported a time to find a ballot in an already opened container but in a different folder. However, since we are assuming that each batch is contained within a container, we take the minimum time available.

Section 5.5 discusses the validity of this assumption in greater detail.

$$W(S) \equiv \sum_{p \in P^S} [s(p) + o(p) + \sum_{b \in B_p^S} (h(b) + t(b))] \tag{5.5}$$

Our workload equation depends on the set of ballots B, the set of batches P, and the specific sampling function for the audit. It measures the amount of work performed in audit A over the batches that are selected for the audit in sample S, P_A, as well as the ballots in those batches B_P. Note that the inner summation is performed up to the last ballot in the batch that is in the sample. For batch audits, this will be the last ballot in the batch. For ballot-level audits, this will be the highest-index ballot that is to be sampled. For example, in a batch of 100 ballots, if ballots 7, 15, and 56 were being sampled, the summation would count up to 56.

5.3.3 Estimating Workload

S is a necessary input for this function. However, S cannot be known for any particular audit a priori, as it is typically generated using a random seed that is not determined until the audit. Fortunately, most extant RLA techniques have sample-size estimation functions that can be used to get a sense of how many and which ballots will be audited. Using these estimation functions, we can assume some average characteristics about the samples that each technique will draw:

1. Batch draws will on average be from the middle of the set of batches if drawn at uniform random.

2. Ballot draws will also on average be from the middle of the batch of ballots.

3. Batch sizes may vary significantly from jurisdiction to jurisdiction, as will the distribution of their sizes. Some jurisdictions may have one large batch and several smaller ones, or roughly even sized batches, etc. Precinct sizes are usually regulated so as to be roughly similar to each other, however small variations may still be present (see [165] for example).

The sample-size estimators for the various audit methods typically do not estimate the total number of samples that will be taken in the audit overall, as such calculation would require knowledge of what the samples can contain. Rather, they are estimates for the first round of the audit. For some methods, like ballot and batch comparison, if no discrepancies are found in the sample, these estimates are reliable predictors of how many samples the audit will need. In others, like ballot polling, these estimates may not confirm the outcome in the first round even if the sample reflects a situation very similar to the reported outcome, especially in elections with closer margins. Therefore, for most results we assume the best-case scenario for audits: that the sample drawn in the first round of the audit is sufficient to meet the risk-limit. This assumption is limiting, and is discussed in more detail in Section 5.5.

For workload calculations, we will construct an "average" sample as follows. For ballot-level audits, as the estimators return a number of ballots, we will find the average distribution of batches occupied by those ballots $\overline{P_A}$ and take its size to obtain the number of batches that will be opened on average. If there are $|P|$ batches of uniform size and $|B_A|$ ballots to audit, the expected number of batches to open is the chance that any batch will contain at least one ballot multiplied by the number of batches, as shown Equation 5.6.

$$|\overline{P_A}| = |P| * (1 - (1 - \frac{1}{|P|})^{|B_A|}) \tag{5.6}$$

However, since batches are rarely of the same size, we modify Equation 5.6 by constructing a set of fake batches P^* of size $|B|$. We partition P^* into $|P|$ parts, where each partition p_i^* is of size p_i, the size of the batch it corresponds to. We then assign ballots as in Equations 5.6 to P^*, and map the resulting audited batches P_A^* back into P to give P_A.

We then assume that the ballots are spread evenly in those batches, iterating over a faux list of ballots to be audited and assigning each one to the batch that has the least number of ballots at each step. Finally, we construct our average sample $\overline{S}$.

$$\overline{S} = \{\overline{P_A}, \overline{B_A}\} \tag{5.7}$$

5.3.4 An example workload comparison

To demonstrate how our workload function applies to various audit types, imagine a jurisdiction that has just held an election with 100,000 ballots cast in 100 batches. This jurisdiction performs a ballot-polling risk-limiting audit based on the method described in Lindeman et al. [143], selecting a risk-limit of 10% and drawing ballots using the countdown method. For a margin of 40%, the expected sample size of the audit is 31 ballots. Using Equation 5.6, we see that the expected number of batches to open is 27. Using the method described above, we populate these 27 batches with the first four containing two ballots and the remaining 23 containing one ballot each.

The first two terms of the workload equation $s(p_i)$ and $o(p_i)$, summed over all batches selected for audit, are $15 \times 27 = 405$. The third term, $h(b_i)$, is 77 for batches with one ballot and 144 for batches with two, yielding $77 * 23 + 144 * 4 = 2,387$. The final term $t(b_i)$ is 25 for batches with one ballot and 50 for batches with two, $25 * 23 + 50 * 4 = 775$. Summing these terms, we obtain a workload of $3,972s$, meaning that a ballot polling audit, performed linearly as described above, would take a little over an hour to complete. Table 5.1 shows these calculations for a few other margins of this audit, and audit-specific workload calculations are performed in Section 5.6.

5.4 Parallelizing the work

As shown in Table 5.1, the workload for an audit can expand fairly quickly as the number of audited ballots and batches increases. However, our model in Equation 5.5 is slightly incomplete. Since the contents of one batch have no bearing on the contents of another, it is entirely possible (and in fact desirable) to parallelize the work, such that batches are opened and ballots are audited simultaneously. The amount of speedup achievable by doing this depends on the number of auditors who can work simultaneously, but in the best case (each batch gets audited simultaneously with every other batch), the workload for a given audit is simply the batch that requires the most work. Put another way, a parallelized version of Equation 5.5 is

$$W_{parallel}(S) \equiv \max_{a \in AB} \sum_{p \in P_a^S} [s(p) + o(p) + \sum_{b \in B_p^S} (h(b) + t(b))] \tag{5.8}$$

Margin	$\lvert\overline{S}\rvert$	$\lvert\overline{P_A}\rvert$	$\sum_{i=1}^{\lvert P_A\rvert} s(p_i)$	$\sum_{i=1}^{\lvert P_A\rvert} o(p_i)$	$\sum_{i=1}^{\lvert P_A\rvert}\sum_{i=1}^{\lvert B_P^A\rvert} f(b_i)$	$\sum_{i=1}^{\lvert P_A\rvert}\sum_{i=1}^{\lvert B_P^A\rvert} t(b_i)$	$W(\overline{S})$
40%	31	27	405	405	2,387	775	3,972
20%	119	70	1,050	1,050	9,163	2,975	14,238
10%	470	100	1,500	1,500	36,190	11,750	50,940
5%	1,861	100	1,500	1,500	143,297	46,525	192,822
1%	46,151	100	1,500	1,500	3,553,627	1,153,775	4,710,402

Table 5.1: **Workload Calculations for a Ballot-Polling Audit, in Seconds**—This table shows the workload calculations for a ballot-polling audit with a 10% risk-limit using the countdown method. Since handling the ballots is the most expensive part of auditing in this way, the $f(b_i)$ term dominates the workload as the number of ballots increases.

Margin	$W(\overline{S})$	$W_{\min}(\overline{S})$	Max Speedup	$W_{10}(\overline{S})$	Speedup with 10 ABs
40%	3,972	234	16.97	498	7.98
20%	14,238	234	60.85	1,434	9.93
10%	50,940	540	94.33	5,094	10.00
5%	192,822	1,968	97.98	19,374	9.95
1%	4,710,402	47,154	99.89	471,132	10.00

Table 5.2: **Parallelization of a Ballot-Polling RLA**—Here we show the workloads achievable if every batch can be opened and its ballots audited simultaneously. Speedup depends on the total number of batches (hence our speedup here approaching 100) as well as the number of available auditing teams. W_{10} shows the workload if ten audit boards are working simultaneously, achieving a speedup of around 10.

$$W_{\min}(S) \equiv \max_{p \in P^S}[s(p) + o(p) + \sum_{b \in B_p^S} (h(b) + t(b))] \tag{5.9}$$

This can dramatically reduce the workload for an audit. Table 5.2 contains parallelized workload results for the example in Section 5.3.4, showing that the speedup in the work is linear in the number of auditors simultaneously auditing batches and ballots.

5.5 Validating the Model

Given that our model has made several assumptions, we now seek to validate it using data from other RLAs.

Colorado The state of Colorado was the first U. S. state to roll out RLAs statewide in 2017. The

Audit	$\lvert S \rvert$	Audit Boards	$\lvert P_A \rvert$	Expected Workload	Observed Workload	% Error
St. Joseph County	453	10	26	4,752	6,421	35.1%
Rhode Island	128	2	3	7,476	8,961	19.9%
City of Lansing	375	4	178	10,836	12,041	11.1%

Table 5.3: **Data from December 2019 Pilots**—Here we show data from three ballot-polling pilots conducted in December 2019 in Rhode Island, Lansing, and St. Joseph County Michigan. The error in our model's workload prediction for each of these audits is relatively small, on the order of 20 minutes or so for each audit. This is accounted for by the fact that the model does not include setup times for the audit, lunch breaks, etc.

Colorado Secretary of State has compiled data from their RLAs on their website, including data from five elections that have been audited since [77]. This data includes timestamps for each audited contest on each ballot as it was evaluated, 427,496 timestamps for audited contests in total. This data is only available for Colorado's ballot comparison audit, which is the technique used for the vast majority of Colorado counties.

The timestamps in this data set only refer to when information for a ballot was entered for comparison. However, we can still use this data to construct durations, by taking the difference in the timestamps for two successively audited ballots. This time will include all steps of our workflow above, from locating and opening the container to recording ballot data. The mean time per contest per ballot is 133 seconds, with a median time of 65 seconds. Our model sees a time of 86 seconds for auditing a ballot in a new batch, which fits roughly in this range. For ballots in an already-opened batch, our model counts 56 seconds per ballot. Therefore, our model appears to be within range of real world data, even in a different jurisdiction, for ballot comparison audits.

Michigan The state of Michigan began piloting risk-limiting audits in 2018, relying almost exclusively on ballot-polling audits [157]. While data from these audits are not publicly available, we obtained timing information collected at four recent Michigan audits in the Muskegon County, Washtenaw Independent School District, St. Joseph County, and the City of Lansing. In total, we obtained 294 measurements corresponding to the duration between when an audit board started pulling ballots from a batch and when they finished using the countdown method. Michigan saw a mean time of 79s for pulling ballots in a ballot-polling audit, which is very close to the 77s reported by RIRLA.

We have included data for two of these audits in Table 5.3. The other two were excluded due to the fact that we did not have end-to-end times recorded for them, only per-ballot measurements. While the model has error on the order of about half an hour in these audits, it is strikingly accurate considering it was developed using data from a completely different jurisdiction. Furthermore, the differences are likely due to unaccounted delays in the auditing process, like lunch breaks or the time it takes to print out and distribute ballot retrieval lists.

Rhode Island Rhode Island conducted another set of RLAs in December of 2019 [190], where they performed a ballot-polling and batch-polling audit. We were able to obtain the total duration, the sample sizes, and the ballot retrieval list for the ballot polling audit. With this information, we ran our model based on the sample that was taken. Table 5.3 displays the data from the ballot-polling audit.

Based on available data, it appears that our model provides a reasonable picture of audits in the real world. Now that we have validated it, we turn our attention to using it to examine extant risk-limiting audit schemes.

5.6 Comparing RLA Techniques

Now that we have a validated RLA model, we turn our attention to examining some commonly used RLA methods. For now, we limit our discussion to the three types of RLA that have been piloted in Rhode Island, and note that these have also been used in Colorado, Michigan, and many other places as well. These techniques are the ballot-polling and ballot comparison audits presented by Lindeman et al. [143], and the batch comparison technique presented by Stark [216].

We examine how the workload changes for each of these three methods in Figure 5.1. As the figure shows, the parallelization of the audit methods is strictly linear, largely preserving proportions. Batch is the only method to have its shape changed, because the workload of auditing batches in parallel is just the workload of auditing one batch, whereas in a linear audit the workloads for each

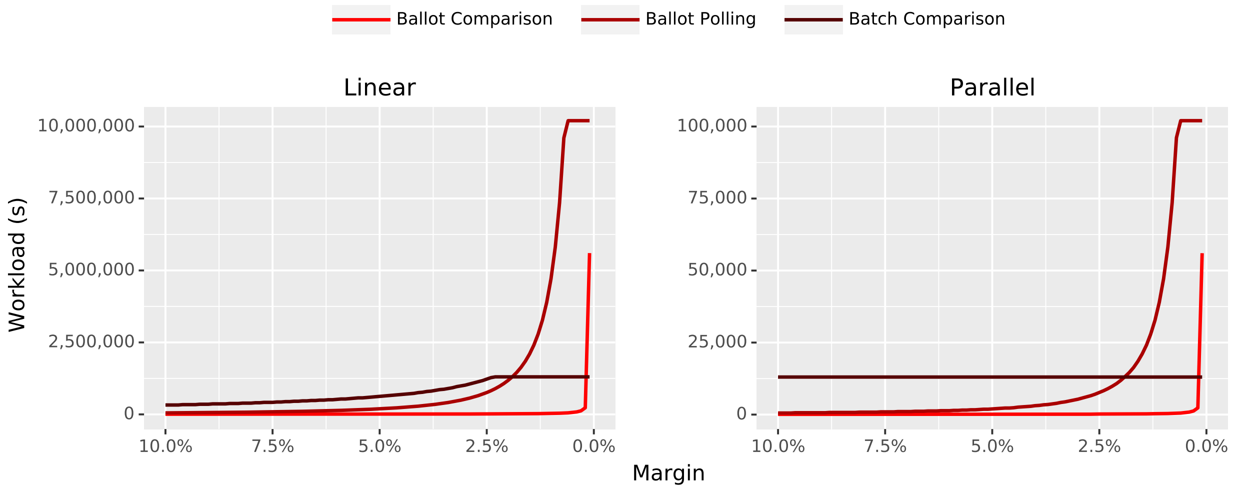


Figure 5.1: *Workload estimates for three types of RLA*—The workloads for ballot-polling, ballot comparison, and batch comparison RLAs with a risk-limit of 10%, 100,000 ballots cast, and 1,000 ballots per batch, by margin. The graphs on the left and the right show the linear and parallel workloads, respectively. The key differences are the scale, and that batch comparison is completely flat in parallelized audits, since all batches are treated equally.

of the batches sum.

Unsurprisingly, ballot comparison is far and away the most efficient method, even with relatively small margins. Ballot polling also does not perform too poorly, although it degenerates much more rapidly than the other two methods as the margins get smaller. Figure 5.2 shows that for margins in the 2 to 3% range and below, ballot-polling audits actually are less efficient than batch comparison. This is because the per-ballot workload of auditing a batch is much lower than the other two methods, although a lot more ballots are audited by batch earlier on. Once ballot polling starts to examine many more ballots, the cost of sorting through the stacks of ballots to find each ballot becomes severe. Due to this finding, it is safe to say that if election officials are choosing between a ballot-polling or batch comparison audit, they should choose a ballot-polling audit only if the true margin is likely to be more than 2 or 3%, depending on the risk-limit. Otherwise, batch comparison, by that point a full hand recount, is more efficient.

Notably, ballot comparison audits also eventually succumb to the inefficiency of random sampling, but margins must be below 0.18% before that occurs. Likewise with our ballot-polling

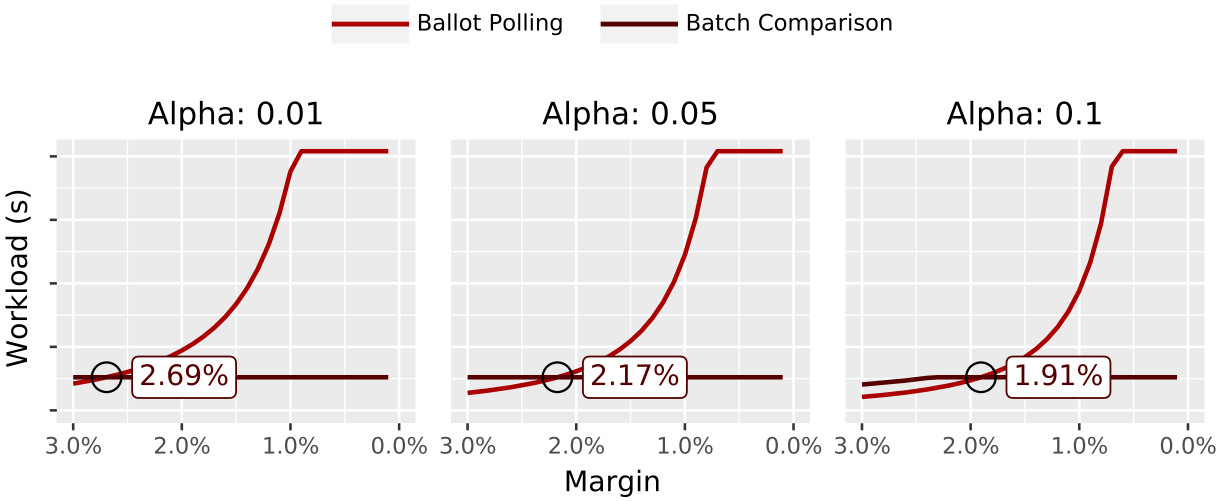


Figure 5.2: *Crossover point for batch comparison and ballot polling*— Ballot-polling audits become less efficient than batch comparison audits when the margins dip below 2 to 3% in elections with 100,000 ballots cast and a batch size of 1,000, depending on the risk-limit as depicted here.

rule of thumb, margins this small are better off just doing a full hand recount.

5.6.1 Election Size

As RLAs rely solely on the margin of an election, and not its size, one might expect that the efficiency would scale with the size of the election. However, as shown in Figure 5.3, this is not quite so. The smaller an election gets, the more efficient batch comparison becomes in contrast to ballot-level methods, because it is so much more efficient at counting ballots. At the other end of the spectrum, the larger an election gets, the more efficient ballot comparison and ballot-polling audits become, because their sample sizes become a smaller fraction of the overall ballots at a faster rate than the batch audit scales. Because batch audits become full recounts always at a fixed point, the ballot-level audits can make up ground. This finding has implications for the types of election audit policies put into practice: it may not be worth it to try to do an RLA of a small election or in a small jurisdiction; however, it will always be worth performing an RLA at larger scales.

Interestingly, the size of the election also has an impact on when a ballot-polling audit becomes less efficient than a batch comparison audit. For large elections, the crossover point is highly dependent on batch size, and can range from as much as 5% down to less than 1%.

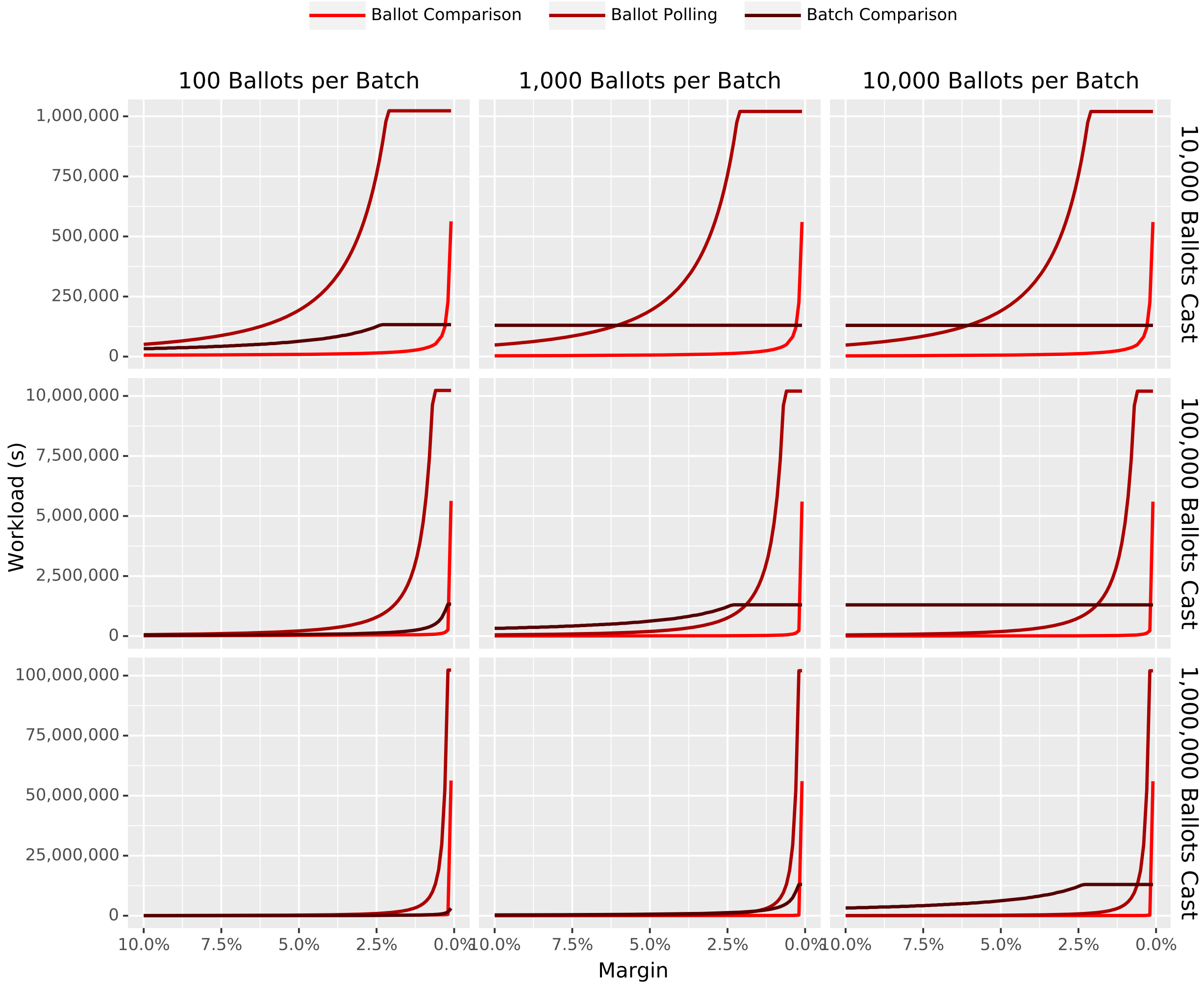


Figure 5.3: *The effect of election and batch size on efficiency*—The relative workloads for ballot-polling, ballot comparison, and batch comparison RLAs vary widely depending on the size of the election (shown here with a risk-limit of 10%). For smaller elections, batch comparison beats out ballot polling with relatively wide margins. The efficiency of ballot comparison audits scales well, however, and its sample sizes and therefore workloads are relatively small regardless of election size. Batch sizing also has essentially no effect on ballot-polling and -comparison audits, but smaller batch sizes do make batch compraison more efficient.

5.6.2 Batch Size

Figure 5.3 also depicts the effect that batch sizing can have on an RLA. Perhaps unsurprisingly, the more batches there are, the more efficient batch comparison audits become. As this was largely the insight that led to ballot-level audits, it is good that our model accurately captures this behavior.

Perhaps more interestingly, batch size does not appear to have much effect on ballot-level audits.

This is likely due to the fact that the times to retrieve and open batches, at a per-ballot level, are strongly dominated by the time it takes to find a ballots. There does not appear to be any advantage, from a workload perspective, of dividing ballots into more or fewer batches. There may be other reasons to batch differently, for example splitting a batch up may make it easier for chain of custody as a smaller stack of ballots gets moved around the audit, or the front end cost of splitting batches may not make it worth it to have smaller batches.

5.6.3 Escalation

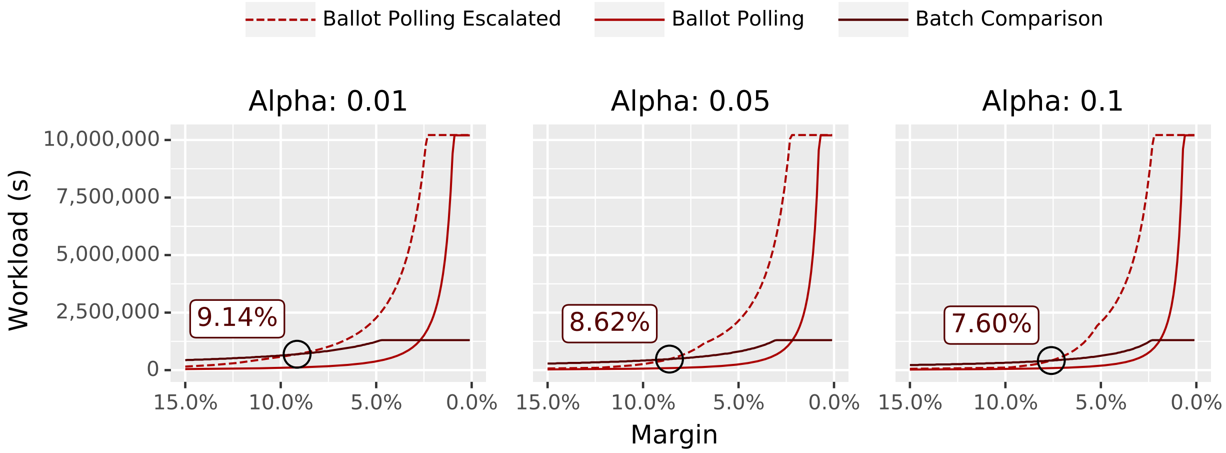


Figure 5.4: *Escalation of a ballot-polling RLA*—Escalation can cause the workload for ballot-polling to explode rather quickly, even in elections with wide margins. Here we depict the additional workload of escalating a sample if the first round sample shows a tie between the winner and the runner up, in an election with 100,000 ballots and a batch-size of 1,000. Interestingly, even for elections with wider margins, it becomes much more efficient to switch to a batch-comparison audit rather than continue ballot polling.

One of the key features of RLAs is that they can escalate their samples based on available evidence. The degree to which escalation impacts the overall workload can be severe, as in the case with a ballot-comparison audit, where even a handful of misstatements will require the next round of the audit to perform a full hand recount. Ballot-polling audits are more forgiving, however there are still strong penalties for not finding sufficient evidence in one round. Indeed, it may be the case frequently that if the audit does not terminate after one round, it is most efficient to switch over to a batch comparison audit than escalate the sample by more ballot polling. Figure 5.4 shows that even in races where margins indicate a tied sample is plausible, the workload explodes to the extent that it becomes less efficient than a batch comparison audit.

5.7 Limitations

In this chapter, we have only examined RLAs through a fairly narrow lens. RLAs are known to break down if more than one contest is targeted for the audit, and this will certainly have an effect on the amount of time it takes to audit each ballot. Because we do not have any data about that, we elected to not attempt to model audits of more than one race. Similarly, more complex elections with more than two candidates, more than one winner, more than one vote allowed in the contest, or election methods other than first-past-the-post also have an impact on the samples that the RLA methods we have examined draw, and therefore the workload analysis.

Another caveat to our model is that it assume that workloads scale linearly. This may not be well founded, as one could conceive of auditors becoming tired and slowing down the rates at which they audit ballots. Conversely, auditors may face a start-up cost of learning or relearning the process, after which they may process ballots more rapidly. This model is only meant to be a jumping off point, and needs further validation against real world data.

Our work has also only concerned itself with three types of RLA, when there are many more. We chose these methods because they are the most commonly adopted, but more recent technologies like SHANGRLA [219] or risk-limiting Bayesian audits [237] may improve the workloads. In particular, these two methods can supported more complex elections, and may also be more efficient in the workloads they produce.

Finally, though we have examined workload in terms of wall-clock time, this is not the only way to account for workload. We chose it because there was available data on which to base a model. In a parallelized audit, though, the cost of running an election likely would not be affected by parallelism, as the same amount of work-per-auditor is being performed. We would present cost estimates, but this is difficult to characterize, as different jurisdictions have different pay scales for different situations. However, since we have evaluated based on time, it should be fairly easy to take an hourly rate of any given jurisdiction and multiply it by our workload estimates to get a ballpark figure.

5.7.1 Future Work

More work is needed to validate this method of analyzing the estimated workload of an audit. For example, in a state like Colorado where two races must always be audited, our estimates may not be accurate without modification. As jurisdictions become more comfortable with auditing and recounting, it is entirely likely that the per-ballot auditing times come down. Finally, we have only focused on time; other dimensions of workload should be examined as well.

5.8 Conclusion

In this chapter, we constructed a model for estimating the workloads of post-election audits. We made several novel observations by applying our model, for example that parallelism really does have a significant effect on the amount of real-world time an audit can take, that ballot-polling audits are only really efficient in specific circumstances, and that escalation has a significant effect on the efficiency of an audit. In the next chapter, we will apply one of these lessons to a real-world setting, and examine how to make an audit parallelizable to the fullest extent possible in a live election environment.

CHAPTER VI

A Manifest Improvement on an Existing RLA Technique[1]

6.1 Introduction

As RLAs are adopted more widely, it is helpful to have methods suitable for different equipment and different election logistics. For instance, jurisdictions vary widely in whether votes are predominantly cast in person or through the mail, in how (and how well) they secure and track physical ballots, and in whether their equipment creates and exports data about how it interpreted individual ballots.

Ballots to be manually checked are drawn in a simple random sample, with or without replacement. Extant methods for both ballot-level comparison audits and ballot-polling audits require a *ballot manifest*, a description of how the physical ballots are organized and stored, so that ballots can be put in a canonical order. With a ballot manifest, it makes sense to say, for instance, "retrieve and inspect the 127,954th ballot cast in the contest." The ballot manifest determines the sampling frame.

As we have just seen, sampling at random is a laborious task, and frequently requires hundreds if not thousands of man-hours to do so for close election margins. Sampling in parallel offers a significant improvement in terms of real-time election auditing, which may be especially salient in elections with tight certification deadlines or suspicious outcomes. While improving the speed of an

[1] This chapter is based on "Bernoulli Ballot-Polling: A Manifest Improvement for Risk-Limiting Audits" work in conjunction with Kellie Ottoboni, J. Alex Halderman, Ronald L. Rivest, and Philip B. Stark that appeared in *Proceedings of the 4th Annual Workshop on Advances in Secure Electronic Voting* [175]

audit in real-time is certainly desirable, it has limits. Most RLA methods require a full accounting of all ballots before the audit can begin, which limits the amount of real-time saving that can be had to after the manifest is prepared. Fully parallelizing an audit is also resource intensive and requires a significant amount of coordination among a large group of people who serve as audit board members. In order to fully take advantage of the parallelism potential in RLAs, these constraints must be reckoned with.

In this chapter, we present an RLA method based on *Bernoulli random sampling*. With simple random sampling, the number of ballots to sample is fixed; with Bernoulli sampling, the *expected sampling rate* is fixed but the sample size is not. Conceptually, *Bernoulli ballot polling* (BBP) decides whether to include the jth ballot in the sample by tossing a biased coin that has probability p of landing heads. The ballot is included if and only if the coin lands heads. Coin tosses for different ballots are independent, but have the same chance of landing heads. (Rather than toss a coin for each ballot, it more efficient to implement Bernoulli sampling in practice using *geometric skipping*, described in Section 6.6.3.)

The logistical simplicity of Bernoulli sampling may make it useful for election audits. Like all RLAs, BBP RLAs require a voter-verifiable paper record. Like other ballot-polling RLAs [142, 143], BBP makes no other technical demands on the voting system. It requires no special equipment, and only a minimal amount of software to select and analyze the sample—in principle, it could be carried out with dice and a pencil and paper. In contrast to extant ballot-polling RLAs, BBP does *not* require a ballot manifest (although it does require knowing where all the ballots are, and access to the ballots). BBP is inherently local and parallelizable, because the decision of whether to include any particular ballot in the sample does not depend on which other ballots are selected, nor on how many other ballots have been selected, nor even on how many ballots were cast. We shall see that this has practical advantages.

Bernoulli sampling is well-known in the survey sampling literature, but it is used less often than simple random sampling for a number of reasons. The variance of estimates based on Bernoulli samples tends to be larger than for simple random samples [201], due to the fact that both the sample and the sample size are random. This added randomness complicates rigorous inferences. A common estimator of the population mean from a Bernoulli sample is the Horvitz-Thompson estimator, which has a high variance when the sampling rate p is small. Often, P-values and confidence intervals for the Horvitz-Thompson estimator are approximated using the normal distribution [75, 146, 222], which may be inaccurate if the population distribution is skewed—as it often is in auditing problems [177].

Instead of relying on parametric approximations, we develop a test based on Wald's sequential probability ratio test [239]. The test is akin to that in extant ballot polling RLA methods [142, 143], but the mathematics are modified to work with Bernoulli random samples, including the fact that Bernoulli samples are drawn without replacement. (Previous ballot-polling RLAs relied on sampling with replacement.) Conditional on the attained sample size n, a Bernoulli sample of ballots is a simple random sample. We maximize the conditional P-value of the null hypothesis (that the reported winner did not win) over a nuisance parameter, the total number of ballots with valid votes for either of a given pair of candidates, excluding invalid ballots or ballots for other candidates. A martingale argument shows that the resulting test is sequential: if the test does not reject, the sample can be expanded using additional rounds of Bernoulli sampling (with the same or different expected sampling rates) and the resulting P-values will still be conservative.

A BBP RLA can begin in polling places on election night. Given an initial sampling rate to be used across all precincts and vote centers, poll workers in each location determine which ballots will be examined in the audit, independently from each other and independently across ballots, and record the votes cast on each ballot selected. (Vote-by-mail and provisional ballots can be audited similarly; see Section 6.6.2.) Once the election results are reported, the sequential probability ratio test can be applied to the sample vote tallies to determine whether there is sufficient evidence that the reported outcome is correct.[2] If the sample does not provide sufficiently strong evidence to attain the risk limit, the sample can be expanded using subsequent rounds of Bernoulli sampling until either the risk limit is attained or all ballots are inspected. Figure 6.1 summarizes the procedure.

[2] The current method uses the reported results to construct the alternative hypothesis. A variant of the method does not require the reported results. We do not present that method here; it is related to ClipAudit [192].

BBP has a number of practical advantages, with little additional workload in terms of the number of ballots examined. Workload simulations show that the number of ballots needed to confirm a correctly reported outcome is similar for BBP and the BRAVO RLA [142]. If the choice of initial sampling rate (and thus, the initial sample size) is larger than necessary, the added efficiency of conducting the audit "in parallel" across the entire election may outweigh the cost of examining extra ballots. Using statewide results from the 2016 United States presidential election, BBP with a 1% initial sampling rate would have had at least a 99% chance of confirming the results in 42 states (assuming the reported results were in fact correct). A Python implementation of BBP is available at https://github.com/pbstark/BernoulliBallotPolling.

Procedure for a Bernoulli ballot-polling audit

1. **Set initial sampling rate.** Choose initial sampling rate p_0 based on pre-election polls or set at a fixed value. If p_0 is selected based on an estimated margin, use the ASN heuristic in Section 6.5.
2. **Sample ballots and record audit data.** Use geometric skipping (below) with rate p_0 to select ballots to inspect. Record votes on all inspected ballots.
3. **Check attained risk.** Once the final election results have been reported, for each contest under audit and for each reported (winner, loser) pair (w, ℓ):
    - Calculate B_w, B_ℓ, and B_u from the audit sample.
    - Find the (maximal) P-value from B_w, B_ℓ, B_u using the test in Section 6.3.
4. **Escalate if necessary.** If, for any (w, ℓ) pair, the P-value is greater than α, expand the audit in one of the ways described in Section 6.4.

Procedure for geometric skip sampling

1. **Set the random seed.** In each polling place, use a cryptographically secure PRNG, such as SHA-256, with a seed chosen using true randomness.
2. **Sample ballots.** Following Section 6.6.3, for each batch of ballots: Set $Y_0 = 0$ and set $j = 0$.
    - $j \leftarrow j + 1$
    - Generate a uniform random variable U on $[0, 1)$.
    - $Y_j \leftarrow \left\lceil \frac{\ln(U)}{\ln(1-p)} \right\rceil$.
    - If $\sum_{k=1}^{j} Y_j$ is greater than the number of ballots in the batch, stop. Otherwise, skip the next $Y_j - 1$ ballots in the batch, and include the ballot after that one (i.e., include ballot $\sum_{k=1}^{j} Y_j$)

Figure 6.1: **Bernoulli ballot-polling audit step-by-step procedures.**

6.2 Notation and Mathematical Background

We consider social choice functions that are variants of majority and plurality voting: the winners are the $k \geq 1$ candidates who receive the most votes. This includes ordinary "first-past-the-post" contests, as well as "vote for k" contests.[3] As explained in [142], it suffices to consider one $(\text{winner}, \text{loser})$ pair at a time: the contest outcome is correct if every reported winner actually received more votes than every reported loser. Auditing majority and super-majority contests requires only minor modifications.[4] Section 6.3.2 addresses auditing multiple contests simultaneously.

Let w denote a reported winning candidate and ℓ denote a reported losing candidate. Suppose that the population contains N_w ballots with a valid vote for w but not ℓ, N_ℓ ballots with a valid vote for ℓ but not w, and N_u ballots with votes for both w and ℓ or for neither w nor ℓ. The total number of ballots is $N = N_w + N_\ell + N_u$. Let $N_{w\ell} \equiv N_w + N_\ell$ be the number of ballots in the population with a valid vote for w or ℓ but not both. For Bernoulli sampling, N may be unknown; in any event, N_w, N_ℓ, and N_u are unknown, or the audit would not be necessary.

If we can reject the null hypothesis that $N_\ell \geq N_w$ at significance level α, we have statistically confirmed that w got more votes than ℓ. Section 6.3 presents a test for this hypothesis that accounts for the nuisance parameter $N_{w\ell}$. We assume that ties are settled in a deterministic way and that if the audit is unable to confirm the contest outcome, a full manual tally resulting in a tie would be settled in the same deterministic way.

6.2.1 Multi-round Bernoulli Sampling

A *Bernoulli*(p) *random variable* $\mathscr{I}$ is a random variable that takes the value 1 with probability p and the value 0 with probability $1 - p$. BBP uses Bernoulli sampling, which involves independent selection of different ballots with the same probability p of selecting each ballot: $\mathscr{I}_j = 1$ if and only if ballot j is selected to be in the sample, where $\{\mathscr{I}_j\}_{j=1}^N$ are independent, identically distributed (IID) Bernoulli(p) random variables.

[3] The same general approach works for some preferential voting schemes, such as Borda count and range voting, and for proportional representation schemes such as D'Hondt [220]. We do not consider instant-runoff voting (IRV).

[4] For instance, for a majority contest, one simply pools the votes for all the reported losers into a single "pseudo-candidate" who reportedly lost.

Suppose that after tossing a coin with probability p_0 of landing heads for every item in the population, we toss a coin with probability p_1 for every item (again, independently), and include an item in the sample if the first or second toss for that item landed heads. That amounts to drawing a Bernoulli sample using selection probability $1-(1-p_0)(1-p_1)$: an item is in the sample unless its coin landed tails on both tosses, which has probability $(1-p_0)(1-p_1)$. This extends to making any integral number K of passes through the population of ballots, with pass k using a coin that has chance p_k of landing heads: such "K-round" Bernoulli sampling is still Bernoulli sampling, with $\mathbb{P}\{\mathscr{I}=1\}=p=1-\prod_{k=0}^{K-1}(1-p_k)$.

6.2.2 Exchangeability and Conditional Simple Random Sampling

Because the N variables $\{\mathscr{I}_j\}$ are IID, they are *exchangeable*, meaning their joint distribution is invariant under the action of the symmetric group (relabelings). Consider a collection of indices $\mathscr{S}\subset\{1,\ldots,N\}$ of size k, $0\le k\le N$. Define the event

$$\mathscr{I}_{\mathscr{S}}\equiv\{\mathscr{I}_j=1,\forall j\in\mathscr{S},\text{ and }\mathscr{I}_j=0,\forall j\notin\mathscr{S}\}.$$

Because $\{\mathscr{I}_j\}$ are exchangeable, $\mathbb{P}\mathscr{I}_{\mathscr{S}}=\mathbb{P}\mathscr{I}_{\mathscr{T}}$ for every set $\mathscr{T}\subset\{1,\ldots,N\}$ of size k, since every such set $\mathscr{T}$ can be mapped to $\mathscr{S}$ by a one-to-one relabeling of the indices.

It follows that, conditional on the attained size of the sample, $n=\sum_{j=1}^{N}\mathscr{I}_j$, all $\binom{N}{n}$ subsets of size n drawn from the N items are equally likely: the sample is conditionally a simple random sample (SRS) of size n. This is foundational for applying the SPRT to Bernoulli samples.

6.3 Tests

Suppose we draw a Bernoulli sample of ballots. The random variable B is the number of ballots in the sample. Let B_w denote the number of ballots in the sample with a vote for w but not ℓ; let B_ℓ denote the number of ballots in the sample with a vote for ℓ but not w; and let B_u denote the number of ballots in the sample with a vote for both w and ℓ or neither w nor ℓ, so $B=B_w+B_\ell+B_u$.

6.3.1 Wald's SPRT with a Nuisance Parameter

We want to test the compound hypothesis that $N_w \leq N_\ell$ against the alternative that $N_w = V_w$, $N_\ell = V_\ell$, and $N_u = V_u$, with $V_w - V_\ell > 0$.[5] We present a test based on Wald's sequential probability ratio test (SPRT) [239].

The values V_w, V_ℓ, and V_u are the reported results (or values related to those reported results; see [142]). In this problem, N_u (equivalently, $N_{w\ell} \equiv N_w + N_\ell$) is a nuisance parameter: we care about $N_w - N_\ell$, the margin of the reported winner over the reported loser.

Conditional on $B = n$, the sample is a simple random sample. The conditional probability that the sample will yield counts (B_w, B_ℓ, B_u) under the alternative hypothesis is

$$\frac{\prod_{i=0}^{B_w-1}(V_w - i)\, \prod_{i=0}^{B_\ell-1}(V_\ell - i)\, \prod_{i=0}^{B_u-1}(V_u - i)}{\prod_{i=0}^{n-1}(N - i)}.$$

If $B_\ell \geq B_w$, the data obviously do not provide evidence against the null, so we suppose that $B_\ell < B_w$, in which case, the element of the null that will maximize the probability of the observed data has $N_w = N_\ell$. Under the null hypothesis, the conditional probability of observing (B_w, B_ℓ, B_u) is

$$\frac{\prod_{i=0}^{B_w-1}(N_w - i)\, \prod_{i=0}^{B_\ell-1}(N_w - i) \prod_{i=0}^{B_u-1}(N_u - i)}{\prod_{i=0}^{n}(N - i)},$$

for some value N_w and the corresponding $N_u = N - 2N_w$. How large can that probability be if the null hypothesis is true? The probability under the null is maximized by any integer $x \in \{\max(B_w, B_\ell), \ldots, (N - B_u)/2\}$ that maximizes

$$\prod_{i=0}^{B_w-1}(x - i) \prod_{i=0}^{B_\ell-1}(x - i) \prod_{i=0}^{B_u-1}(N - 2x - i).$$

[5] The alternative hypothesis is that the reported results are correct; as mentioned above, there are other approaches one could use that do not involve the reported results, but we do not present them here.

The logarithm is monotonic, so any maximizer x^* also maximizes

$$f(x) = \sum_{i=0}^{B_w-1} \ln(x-i) + \sum_{i=0}^{B_\ell-1} \ln(x-i) + \sum_{i=0}^{B_u-1} \ln(N-2x-i).$$

The second derivative of f is everywhere negative, so f is convex and has a unique real-valued maximizer on $[\max(B_w, B_\ell), (N-B_u)/2]$, either at one of the endpoints or somewhere in the interval. The derivative $f'(x)$ is

$$f'(x) = \sum_{i=0}^{B_w-1} \frac{1}{x-i} + \sum_{i=0}^{B_\ell-1} \frac{1}{x-i} - 2\sum_{i=0}^{B_u-1} \frac{1}{N-2x-i}.$$

If $f'(x)$ does not change signs, then the maximum is at one of the endpoints, in which case x^* is the endpoint for which f is larger. Otherwise, the real maximizer occurs at a stationary point. If the real-valued maximizer is not an integer, convexity guarantees that the integer maximizer x^* is one of the two integer values that bracket the real maximizer: either $\lfloor x \rfloor$ or $\lceil x \rceil$.

A conservative P-value for the null hypothesis after n items have been drawn is thus

$$P_n = \frac{\prod_{i=0}^{B_w-1}(x^*-i) \prod_{i=0}^{B_\ell-1}(x^*-i) \prod_{i=0}^{B_u-1}(N-2x^*-i)}{\prod_{i=0}^{B_w-1}(V_w-i) \prod_{i=0}^{B_\ell-1}(V_\ell-i) \prod_{i=0}^{B_u-1}(V_u-i)}.$$

Wald's SPRT [239] leads to an elegant escalation method if the first round of Bernoulli sampling does not attain the risk limit: simply make another round of Bernoulli sampling, as described in Section 6.4. If the null hypothesis is true, then $\Pr\{\inf_k P_k < \alpha\} \leq \alpha$, where k counts the rounds of Bernoulli sampling. That is, the risk limit remains conservative for any number of rounds of Bernoulli sampling.

6.3.2 Auditing Multiple Contests

The math extends to audits of multiple contests; we omit the derivation, but see, e.g., [143]. The same sample can be used to audit any number of contests simultaneously. The audit proceeds to a full hand count unless every null hypothesis is rejected, that is, unless we conclude that *every* winner beat *every* loser in *every* audited contest. The chance of rejecting all those null hypotheses cannot be larger than the smallest chance of rejecting any of the individual hypotheses, because the probability of an intersection of events cannot be larger than the probability of any one of the events. The chance of rejecting any individual null hypothesis is at most the risk limit, α, if that hypothesis is true. Therefore the chance of the intersection is not larger than α if any contest outcome is incorrect: the overall risk limit is α, with no need to adjust for multiplicity.

6.4 Escalation

If the first round of Bernoulli sampling with rate p_0 does not generate strong evidence that the election outcome is correct, we have several options:

1. conduct a full hand count
2. augment the sample with additional ballots selected in some manner, for instance, making additional rounds of Bernoulli sampling, possibly with different values of p
3. draw a new sample and use a different auditing method, *e.g.*, ballot-level comparison auditing

The first approach is always conservative. Both the second and third approaches require some statistical care, as repeated testing introduces additional opportunities to wrongly conclude that an incorrect election outcome is correct.

To make additional rounds of Bernoulli sampling, it may help to keep track of which ballots have been inspected.[6] That might involve stamping audited ballots with "audited" in red ink, for example.

Section 6.2.1 shows that if we make an integral number of passes through the population of ballots, tossing a p_k-coin for each as-yet-unselected item (we only toss the coin for an item on the kth pass if the coin has not landed heads for that item in any previous pass), then the resulting sample is a Bernoulli random sample with selection probability $p = 1 - \prod_{k=0}^{K-1}(1 - p_k)$. Conditional on the sample size n attained after K passes, every subset of size n is equally likely to be selected. Hence, the sample is conditionally a simple random sample of size n from the N ballots.

[6] Once ballots are aggregated in a precinct or scanned centrally, it is unlikely that they will stay in the same order.

The SPRT applied to multi-round Bernoulli sampling is conservative: the unconditional chance of rejecting the null hypothesis if it is true is at most α, because, if the null is true, the chance that the SPRT exceeds $1/\alpha$ for *any* K is at most α.

The third approach allows us to follow BBP with a different, more efficient approach, such as ballot-level comparison auditing [143]. This may require steps to ensure that multiplicity does not make the risk larger than the nominal risk limit, e.g., by adjusting the risk limit using Bonferroni's inequality.

6.5 Initial Sampling Rate

We would like to choose the initial sampling rate p_0 sufficiently large that a test of the hypothesis $N_w \leq N_\ell$ will have high power against the alternative $N_w = V_w, N_\ell = V_\ell$, with $V_w - V_\ell = c$ for modest margins $c > 0$, but not so large that we waste effort.

There is no analytical formula for the power of the sequential hypothesis test under this sampling procedure, but we can use simulation to estimate the sampling rates needed to have a high probability of confirming correctly reported election results. Table 6.1 gives the sampling rate p_0 needed to attain 80%, 90%, and 99% power for a 2-candidate race in which there are no undervotes or invalid votes, for a 5% risk limit and a variety of margins and contest sizes. The simulations assume that the reported vote totals are correct. The required p_0 may be prohibitively large for small races and tight margins; Section 6.7 shows that with high probability, even a 1% sampling rate would be sufficient to confirm the outcomes of the vast majority of U.S. federal races without further escalation.

The sequential probability ratio test in Section 6.3 is similar to the BRAVO RLA presented in [143] when the sampling rate is small relative to the population size. There are two differences between BRAVO and BBP: BBP incorporates information about the number of undervotes, invalid votes, or votes for candidates other than w and ℓ, and Bernoulli sampling is done without (as opposed to with) replacement. If every ballot has a valid vote either for w or for ℓ and the sampling rate is small relative to the population size, the expected workload of these two procedures is similar. The *average sample number* (ASN) [239], the expected number of draws required either to accept or to

Table 6.1: **Estimated sampling rates needed for Bernoulli ballot polling** for a 2-candidate race with a 5% risk limit. These simulations assume the reported margins were correct.

		sampling rate p to achieve …		
true margin	ballots cast	80% power	90% power	99% power
1%	100,000	55%	62%	77%
2%	100,000	23%	30%	46%
5%	100,000	5%	7%	12%
10%	100,000	2%	2%	4%
20%	100,000	1%	1%	1%
1%	1,000,000	10.4%	14.2%	24.2%
2%	1,000,000	2.9%	4.0%	7.5%
5%	1,000,000	0.5%	0.7%	1.3%
10%	1,000,000	0.2%	0.2%	0.4%
20%	1,000,000	0.1%	0.1%	0.1%
1%	10,000,000	1.15%	1.66%	3.11%
2%	10,000,000	0.30%	0.42%	0.84%
5%	10,000,000	0.05%	0.07%	0.13%
10%	10,000,000	0.02%	0.02%	0.04%
20%	10,000,000	0.01%	0.01%	0.01%

reject the null hypothesis, for BRAVO using a risk limit α and margin m is approximately

$$\text{ASN} \approx \frac{2\ln(1/\alpha)}{m^2}.$$

This formula is valid when the sampling rate is low and the actual margin is not substantially smaller than the (reported) margin used as the alternative hypothesis.

The ASN gives a rule of thumb for choosing the initial sampling rate for BBP. For a risk limit of 5% and a margin of 5%, the ASN is about 2,400 ballots. For a margin of 10%, the ASN is about 600 ballots. These values are lower than the sample sizes implied by Table 6.1: the sampling rates in the table have a higher probability that the initial sample will be sufficient to conclude the audit, while a sampling rate based on the ASN will suffice a bit more than half of the time.[7] The ASN multiplied by 2–4 is a rough approximation to initial sample size needed to have roughly a 90%

[7] The distribution of the sample size is skewed to the right: the expected sample size is generally larger than the median sample size.

chance that the audit can stop without additional sampling, if the reported results are correct.

The ASN formula assumes that N_u is 0; value of p_0 should be adjusted to account for ballots that have votes for neither w nor ℓ (or for both w and ℓ). If $r = \frac{N_u}{N}$ is the fraction of such ballots, the initial sampling rate p_0 should be inflated by a factor of $\frac{1}{1-r}$. For example, if half of the ballots were undervotes or invalid votes, then double the sampling rate would be needed to achieve the same power as if all of the ballots were valid votes for either w or ℓ.

6.6 Implementation

6.6.1 Election Night Auditing

Previous approaches to auditing require a sampling frame (possibly stratified, *e.g.*, by mode of voting or county). That requires knowing how many ballots were cast and their locations. In contrast, Bernoulli sampling makes it possible to start the audit at polling places immediately after the last vote has been cast in that polling place, without even having to count the ballots cast in the polling place. This has several advantages:

1. It parallelizes the auditing task and can take advantage of staff (and observers) who are already on site at polling places.

2. It takes place earlier in the chain of custody of the physical ballots, before the ballots are exposed to some risks of loss, addition, substitution, or alteration.

3. It may add confidence to election-night result reporting.

The benefit is largest if p_0 is large enough to allow the audit to complete without escalating. Since reported margins will not be known on election night, p_0 might be based on pre-election polls, or set to a fixed value. There is, of course, a chance that the initial sample will not suffice to confirm outcomes, either because the true margins are smaller than anticipated, or because the election outcome is in fact incorrect.

There are reasons polling-place BBP audits might not be desirable.

1. Pollworkers, election judges, and observers are likely to be tired and ready to go home when polls close.

2. The training required to conduct and to observe the audit goes beyond what poll workers and poll watchers usually receive.

3. Audit data need to be captured and communicated reliably to a central authority to compute the risk (and possibly escalate the audit) after election results are reported.

6.6.2 Vote-by-mail and Provisional Ballots

The fact that Bernoulli sampling is a "streaming" algorithm may help simplify logistics compared with other sampling methods. For instance, Bernoulli sampling can be used with vote-by-mail (VBM) ballots and provisional ballots. VBM and provisional ballots can be sampled as they arrive (after signature verification), or aggregated, e.g., daily or weekly. Ballots do not need to be opened or examined immediately in order to be included in the sample: they can be set aside and inspected after election day or after their provisional status has been adjudicated. Any of these approaches yields a Bernoulli sample of all ballots cast in the election, provided the same value(s) of p are used throughout.

6.6.3 Geometric Skipping

In principle, one can implement Bernoulli sampling by actually rolling dice, or by assigning a $U[0,1]$ random number to each ballot, independently across ballots. A ballot is in the sample if and only if its associated random number is less than or equal to p.

However, that places an unnecessarily high burden on the quality of the pseudorandom number generator—or on the patience of the people responsible for selecting ballots by mechanical means, such as by rolling dice. If the ballots are in physical groups (e.g., all ballots cast in a precinct), it can be more efficient to put the ballots into some canonical order (for instance, the order in which they are bundled or stacked) and to rely on the fact that the *waiting times* between successes in independent Bernoulli(p) trials are independent Geometric(p) random variables: the chance that the next time the coin lands heads will be kth tosses after the current toss is $p(1-p)^{k-1}$.

To select the sample, instead of generating a Bernoulli random variable for every ballot, we suggest generating a sequence of geometric random variables $Y_1, Y_2, \ldots$ The first ballot in the sample is the one in position Y_1 in the group, the second is the one in position $Y_1 + Y_2$, and so on. We continue in this way until $Y_1 + \cdots + Y_j$ is larger than the number of ballots in the group. This *geometric skipping* method is implemented in the software we provide.

6.6.4 Pseudorandom Number Generation

To draw the sample, we propose using a cryptographically secure PRNG based on the SHA-256 hash function, setting the seed using 20 rolls of 10-sided dice, in a public ceremony. This is the method that the State of Colorado uses to select the sample for risk-limiting audits.

This is a good choice for election audits for several reasons. First, given the initial seed, anyone can verify that the sequence of ballots audited is correct. Second, unless the seed is known, the ballots to be audited are unpredictable, making it difficult for an adversary to "game" the audit. Finally, this family of PRNGs produces high-quality pseudorandomness.

Implementations of SHA-256-based PRNGs are available in many languages, including Python and Javascript. The code we provide for geometric skipping relies on the `cryptorandom` Python library, which implements such a PRNG.

While Colorado sets the seed for the entire state in a public ceremony, it may be more secure to generate seeds for polling-place audits locally, after the ballots have been collated into stacks that determine their order for the purpose of the audit. If the seed were known before the order of the ballots was fixed, an adversary might be able to arrange that the ballots selected for auditing reflect a dishonest outcome.

While the sequence of ballots selected by this method is verifiable, there is no obvious way to verify *post facto* that the ballots examined were the correct ones. Only observers of the audit can verify that. Observers' job would be easier if ballots were pre-stamped with (known) unique identifiers, but that might compromise vote anonymity.

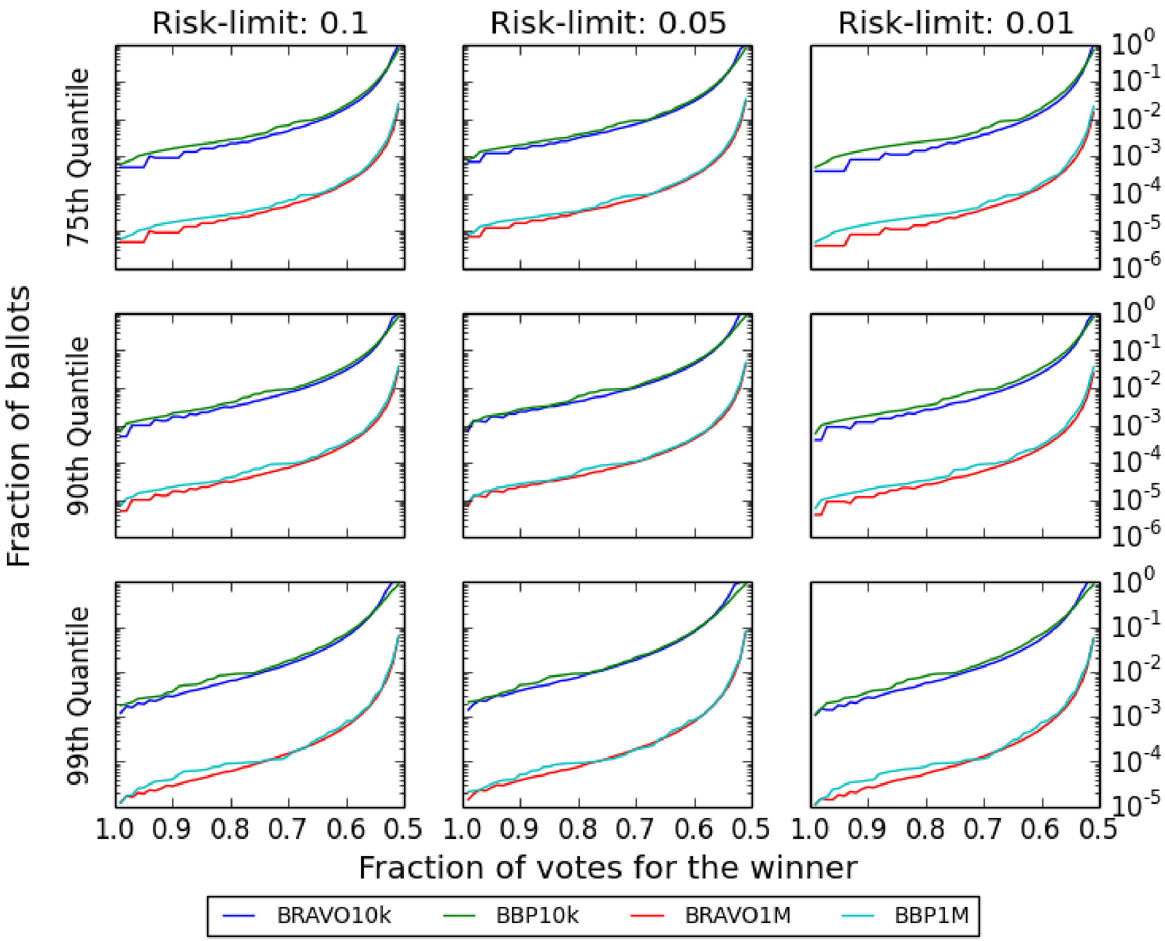


Figure 6.2: **Simulated quantiles of sample sizes by fraction of votes for the winner for a two candidate race** in elections with 10,000 ballots and 1 million ballots, for BRAVO ballot-polling audits (BPA) and Bernoulli ballot polling audits (BBP), for various risk-limits. The simulations assume every ballot has a valid vote for one of the two candidates.

6.7 Evaluation

As discussed in Section 6.5, we expect that *workload* (total number of ballots examined) for Bernoulli ballot polling to be approximately the same as BRAVO ballot polling. Figure 6.2 compares the fraction of ballots examined for BRAVO audits and BBP for a 2-candidate contest, estimated by simulation. The simulations use contest sizes of 10,000 and 1,000,000 ballots, each of which has either a valid vote for the winner or a valid vote for the loser. The percentage of votes for the winner ranges from 99% (almost all the votes go to the winner) to 50% (a tie). The methods produce similarly shaped curves; BBP requires slightly more ballots than BRAVO.

As the workload of BRAVO and BBP are similar, the cost of running a Bernoulli audit should be similar to BRAVO. There are likely other efficiencies to Bernoulli audits, *e.g.*, if the first stage of the audit can be completed on election night in parallel, it might result in lower cost as election workers and observers would not have to assemble in a different place and time for the audit. Even if the cost were somewhat higher, that might be offset by advantages discussed in Section 6.8.

6.7.1 Empirical Data

We evaluate BBP using precinct-level data from the 2016 U.S. presidential election, collected from OpenElections[8] or by hand where that dataset was incomplete. If the reported margins are correct, BBP with a sampling rate of $p_0 = 1\%$ and a risk-limit of 5% would have a 99% or higher chance of confirming the outcome in 42 states. The mean sample size per-precinct for this method is about 10 ballots, indicating that if the audit is conducted in-precinct the workload will be fairly minute. There is thus a large probability that if the election outcomes in those states are correct, they would not have to audit additional ballots beyond the initial sample.

6.8 Discussion

Bernoulli ballot polling has a number of practical advantages. We have discussed several throughout the paper, but we review all of them here:

- It reduces the need for a ballot manifest: ballots can be stored in any order, and the number of ballots in a given container or bundle does not need to be known to draw the sample.
- The work can be conducted in parallel across polling places, and can be performed by workers (and observed by members of the public) already in place on election day.
- The same sampling method can be used for polling places, vote centers, VBM, and provisional ballots, without the need to stratify the sample explicitly.
- If the initial sampling rate is adequate, the winners can be confirmed shortly after voting finishes—perhaps even at the same time that results are announced—possibly increasing voter

[8] http://openelections.net/, last visited 8/5/19.

confidence.

- When a predetermined expected sampling rate is used, the labor required can be estimated in advance, assuming escalation is not required. With appropriate parameter choices, escalation can be avoided except in unusually close races, or when the reported outcome is wrong. This helps election officials plan.

- If the sampling rate is selected after the reported margin is known, officials can choose a rate that makes escalation unlikely unless the reported electoral outcome is incorrect.

- The sampling approach is conceptually easy to grasp: toss a coin for each ballot. The audit stops when the sample shows a sufficiently large margin for every winner over every loser, where "sufficiently large" depends on the sample size.

- The approach may have security advantages, since waiting longer to audit would leave more opportunity for the paper ballots to be compromised or misplaced. Workers will need to handle the ballot papers in any case to move them from the ballot boxes into long-term storage.

Officials selecting an auditing method should weigh these advantages against some potential downsides of our approach, particularly when applied in polling places on election night. Poll workers are already very busy, and they may be too tired at the end of the night to conduct the sampling procedure or to do it accurately. When audits are conducted in parallel at local polling places, it is impossible for an individual observer to witness all the simultaneous steps. Moreover, estimating the sample size before margins are known makes it likely that workers will end up sampling more (or fewer) ballots than necessary to achieve the risk limit. While sampling too little can be overcome with escalation, the desire to avoid escalation may make officials err on the side of caution and sample more than predicted to be necessary, further reducing expected efficiency.

6.8.1 Previous Work

Bernoulli sampling is a special case of Poisson sampling, where sampling units are selected independently, but not necessarily with equal probability. Aslam et al. [19] propose a Poisson sampling method in which the probability of selecting a given unit is related to a bound on the error that unit could hide. Their method is not an RLA: it is designed to have a large chance of detecting at least one error if the outcome is incorrect, rather than to limit the risk of certifying an incorrect outcome *per se*.

6.8.2 Stratified Audits

Independent Bernoulli samples from different populations using the same rate still yields a Bernoulli sample of the overall population, so the math presented here can be used without modification to audit contests that cross jurisdictional boundaries. Bernoulli samples from different strata using different rates can be combined using SUITE [176], which can be applied to stratum-wise P-values from any method, including BBP. (This requires minor modifications to the P-value calculations, to test arbitrary hypotheses about the margin in each stratum rather than to test for ties; the derivations in [176] apply, *mutatis mutandis*.) If some ballots are tabulated using technology that makes a more efficient auditing approach possible, such as a ballot-level comparison audit, it may be advantageous to stratify the ballots into groups, sample using Bernoulli sampling in some and a different method in others, and use SUITE to combine the results into an overall RLA.

6.9 Conclusion

Having identified an area for practical improvement with post-election audits, we presented a new ballot-polling RLA based on Bernoulli sampling, relying on Wald's sequential probability ratio test to calculate the risk limit. The new method performs similarly to the BRAVO ballot-polling audit but has several logistical advantages, including that it can be parallelized and conducted on election night, which may reduce cost and increase security. The method easily incorporates VBM and provisional ballots, and may eliminate the need for stratification in many circumstances. Bernoulli ballot-polling with just a 1% sampling rate would have sufficed to confirm the 2016

U.S. Presidential election results in the vast majority of states, if the reported results were correct. The practical benefits and conceptual simplicity of Bernoulli ballot polling may make it simpler to conduct risk-limiting audits in real elections.

CHAPTER VII

Examining Coercion

7.1 Introduction

So far we have explored the nuances of two types of election security technology designed to mitigate the risk that an election's outcome could be compromised by malicious software misprinting or miscounting votes. These two technologies, voter-verified paper and post-election audits, rely on notions of software-independence and cast-as-intended verifiability to provide evidence for the *correctness* of an election outcome. However, as discussed in Chapter II, the threat surface to elections is broad, and there are other types of attacks that impact both an election's correctness and its verifiability, as well as other desirable properties like protecting voters from being coerced. Here we turn our attention to some of these surfaces, and show that even a voting system that correctly deploys voter-verified paper and post-election audits can be subverted if its other areas, like voter authentication, are not protected.

The 2018 midterm elections in the United States saw numerous contentious races come down to the wire as jurisdictions counted and recounted ballots. One of the biggest pressure points was that of absentee ballots: in many states, there was intense litigation over which absentee ballots could be counted in the totals. For example, Georgia saw a federal court overrule the policies in Gwinnett County, where ballots from Asian-American voters were being rejected at significantly higher rates than other voters [94]. Other states like Florida and Arizona saw similar issues [182, 223], and North Carolina even saw an elaborate fraud scheme designed around "ballot harvesting", a process in which voters' ballots are collected before they are filled out, allowing the collectors to vote for whomever they would like [42].

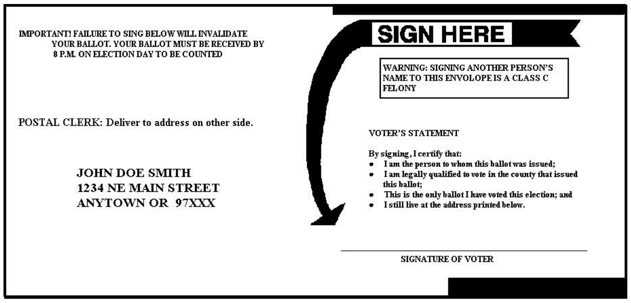
Five states in the U.S. have already moved to an all-vote-by-mail election model [171], and more and more states are looking to ramp up their absentee voting capabilities in the wake of the COVID-19 pandemic [172]. Indeed, states like Georgia and Michigan are already surpassing all-time high turnout in ballots voted by mail [46, 91]. While there has been significant work on remote voting in an Internet context [34, 35, 81], and in particular of coercion and coercion-resistance [134], there is little existing work about vote-by-mail schemes.

Most VBM authentication today is done using voters' wet-ink signatures on their ballot envelopes (in the U.S.) or signature cards (in places like Switzerland [130]). Wet-ink signature validation is often done by hand [163], though there are automated solutions available [207]. Both schemes have inherent flaws, as wet-ink signatures change over time and often are not consistently reproduced by the signer.[1] Moreover, this relatively weak authentication scheme leaves ample room for fraud, either via wet-ink signature forgery, ballot harvesting, or coercion. Wet-ink signatures can also be omitted entirely, by mistake. Some difficulties are well documented in election worker training manuals, such as [167] from Colorado.

Local regulations and election worker training may change between jurisdictions, so absentee ballot rejection rates based on wet-ink signature validation can vary by several percentage points even within one state [209]. All of these factors contribute to a need for a more robust method of voter authentication for absentee ballots. Voter coercion in VBM voting schemes is also not well understood. Anecdotes about coercion abound, as do reports of isolated coercion incidents like those reported in North Carolina in 2018 [42].

In this chapter, we analyze a vote-by-mail scheme based on real-world elections, and explore its security properties in the context of existing threat models like the coercion-resistance model proposed by Juels, Catalano and Jakobsson [125]. We use JCJ as a framework to construct a security model, and use our model to find that existing VBM schemes are severely lacking in two important

[1] The field of forensics has numerous studies about the failure of trained experts and automated systems to correctly identify correct or forged wet-ink signatures, for example [39, 100, 114, 208].



IMPORTANT! FAILURE TO SING BELOW WILL INVALIDATE
YOUR BALLOT. YOUR BALLOT MUST BE RECEIVED BY
8 P.M. ON ELECTION DAY TO BE COUNTED

POSTAL CLERK: Deliver to address on other side.

JOHN DOE SMITH
1234 NE MAIN STREET
ANYTOWN OR 97XXX

SIGN HERE

WARNING: SIGNING ANOTHER PERSON'S NAME TO THIS ENVOLOPE IS A CLASS C FELONY

VOTER'S STATEMENT

By signing, I certify that:

- I am the person to whom this ballot was issued;
- I am legally qualified to vote in the county that issued this ballot;
- This is the only ballot I have voted this election; and
- I still live at the address printed below.

SIGNATURE OF VOTER

Figure 7.1: **Absentee Ballot Envelope from Washington County, Oregon**—An example of the envelope a voter signs when mailing in their ballot [88].

areas: authentication and coercion-resistance. We then examine the effects of policy variations on security, for example whether a jurisdiction allows voters to vote in person and override their mailed-in ballot. We also propose a technical solution based on HMAC-based one time passwords (HOTPs) that replaces wet-ink signatures and solves many of the authentications problems that existing VBM schemes have.

The rest of this chapter is structured as follows: Section 7.2 models VBM as a protocol after [125]. Section 7.3 develops our approach, introducing HOTPs as a replacement for wet-ink signatures and recommending that voters be allowed to revoke a submitted absentee ballot or re-vote. Section 7.4 contextualizes our solution in the current technical and policy environment of U.S. elections, and finally Section 7.5 concludes.

7.2 Modelling Vote-by-Mail

In order to better understand the security properties of VBM, we will construct a framework for VBM after Juels, Catalano, and Jakobsson (JCJ) [125]. While JCJ is not a perfect framework for analysis, and many modifications have to be made to it to fit VBM (which we discuss later), it does provide a good starting point for thinking about secure remote voting schemes. By examining VBM within this framework, we show that it lacks the capability to adequately provide correctness and coercion-resistance properties. Our modified framework also illuminates where improvements can be made, and we use it later on in Section 7.3.3 to evaluate the performance of our improvements.

Remark: Most VBM schemes in the U.S. do not utilize cryptography in the traditional sense, if at all. Here, when we use SK and PK for election authorities, the best analog is the keys used in the certificates authenticating election officials' websites, which are frequently where voter registration takes place [173]. Thus, when we say that a voter roll or tally is signed, we mean that it is published on a website authenticated with TLS. While some election websites do not currently use TLS, they are becoming less and less common due to major pushes by CISA [83], so we will assume for our protocol that TLS is used.

7.2.1 Functions

We adopt the same primitives at the JCJ scheme, including registrars $\mathscr{R}$, talliers $\mathscr{T}$, voters $\mathscr{V}$, candidate slate $\mathscr{C}$, specified by $n_{\mathscr{C}}$, and tally vector $\boldsymbol{X}$. However, as the functions performed in a VBM scheme are slightly different, we modify the four protocols in JCJ as follows and add another protocol, $\mathsf{authenticate}$:

- **Registering**: The function $\mathsf{register}(SK_{\mathscr{R}}, i, \mathsf{sig}_i, \boldsymbol{L}) \rightarrow \{0,1\}$ takes the registrar's secret key, the voter's information i,[2] and the voter's wet-ink signature sig_i, adds the voter to the voter roll $\boldsymbol{L}$ and outputs 1 if the registration was successful, otherwise, outputs 0.

- **Voting**: The function $\mathsf{vote}(\mathsf{sig}, PK_{\mathscr{T}}, n_{\mathscr{C}}, \beta) \rightarrow b$ takes in the voter's wet-ink signature, tallier's public key, the slate of candidates, and the voter's choices and produces a sealed paper ballot b for tallying.

- **Authentication**: The function $\mathsf{authenticate}(\mathsf{sig}, b, \boldsymbol{L}, \mathscr{B}\mathscr{B}_{\text{VBM}}) \rightarrow \{\perp, b_{open}\}$ takes in the voter's wet-ink signature sig, their sealed ballot b, and the list of eligible voters $\boldsymbol{L}$, and determines if sig corresponds to a voter in $\boldsymbol{L}$. If it does, then the ballot is opened and added to the set of ballots $\mathscr{B}$ for tallying, and $\mathscr{B}\mathscr{B}_{\text{VBM}}$ is updated to `True` for the voter. Otherwise, the ballot is discarded.

- **Tallying**: The function $\mathsf{tally}(SK_{\mathscr{T}}, \mathscr{B}, n_{\mathscr{C}}) \rightarrow \boldsymbol{X}$ takes in the tallier's secret key $SK_{\mathscr{T}}$, the set of opened ballots $\mathscr{B}$, and the candidate slate and produces the tally $\boldsymbol{X}$, the vector of summed

[2] Voter information consists of a unique identifier associated with the voter that is maintained in the voter registration database, and is not public.

vote totals for each selection.

- **Verifying**: The function $\mathsf{verify}(PK_{\mathscr{T}}, \mathscr{B}\mathscr{B}_{VBM}, \mathscr{B}, n_{\mathscr{C}}, \mathbf{X}) \rightarrow \{0,1\}$ takes the public key of the talliers $PK_{\mathscr{T}}$, the bulletin board $\mathscr{B}\mathscr{B}_{\mathrm{VBM}}$, the set of tallied ballots $\mathscr{B}$, the candidate slate $n_{\mathscr{C}}$, and the tally $\boldsymbol{X}$ and outputs 1 if the result verifies as correct, 0 otherwise.

7.2.2 Setup

VBM deviates from the JCJ formulation of an election in a few key ways. There are no keys generated or used in the process (except the TLS keys for the registration website and results reporting website), so the setup phase simply consists of publishing the slate of candidates.

Registration is two steps in VBM: voters register to vote and obtain a credential with which to vote (in the form of a wet-ink signature on file in the clerk's office, loosely akin to the secret key in JCJ). Once the voter has mailed in their ballot, the clerk compares the wet-ink signature included with the ballots with the one on file to verify that it matches, and if it does the vote is counted (akin to "permitting participation in the election").

As votes have no commitments when they are submitted, there is also no bulletin board in the traditional sense. This means that tallying simply consists of counting the plaintext paper ballots.

Additionally, the lack of a bulletin board means that that voters cannot individually verify that their ballots were received and correctly included in the tally during the verification phase. However, voters can query clerks to determine that their ballots were received, and there is not usually authentication of the voters identity [166, 168, 169]. We model this as a bulletin board $\mathscr{B}\mathscr{B}_{\mathrm{VBM}}$ consisting of the list of registered voters and a boolean value for each voter. $\mathscr{B}\mathscr{B}_{\mathrm{VBM}}$ is initialized with the list of eligible voters and `False` for every voter, and when ballots are received and authenticated the boolean corresponding with that voter is changed to `True`. Importantly, once a ballot has been received for a voter and $\mathscr{B}\mathscr{B}_{\mathrm{VBM}}$ for that voter is set to `True`, the ballot cannot be revoked. Voters in our model are not allowed to re-vote once they have voted by mail, a constraint that is required by many VBM schemes within the U.S. We consider schemes where voters may mutate their ballot after initial submission outside of scope for now, but we discuss the implications of this later on.

Similarly, the lack of a traditional bulletin board means that the tally cannot be recomputed by summing and decrypting its contents. However, weaker properties, like ensuring the tally corresponds to the number of voters who voted are still verifiable. Furthermore, a post-election audit, like a risk-limiting audit [215] can provide a statistical degree of verifiability, while not achieving full universal verifiability. We will assume that an RLA is performed.

Finally, talliers and registrars are typically the same people: local clerks are responsible for both determining who is eligible to vote as well as counting the ballots after the election. While we distinguish above, $\mathscr{R}$ and $\mathscr{T}$ are identical and could be substituted for one another.

7.2.3 Assumptions

Our setup phase requires no assumptions. Adversaries in VBM may coerce voters prior to the registration phase, requiring that they register and request a ballot. This may include having the voter's ballot mailed to the adversary or registering a signature chosen by the adversary.

The merging of the talliers and registrars constrains the requirement during registration that the adversary cannot corrupt the registrars, since the scheme VBM also fails to provide for the other two registration phase assumptions, leading to a simulation attack as described in JCJ (discussed further below). However, we will assume that the adversary cannot impersonate the voter during the registration phase, only that they can seize their credentials once registered. This is reasonable, as registration typically requires fairly thorough verification of the voter's identity that goes beyond what an adversary may be able to obtain from the voter (see [170] as an example of the information needed).

VBM needs stronger assumptions on the anonymous channel on which voters can cast votes. As with JCJ, we assume the channel faithfully transmits the vote that was submitted to it, i.e. that the postal service is honest. We term this the *honest postal system hypothesis*.

If an attacker has corrupted the talliers, it is possible that ballots will not be adequately separated from their outer envelopes, revealing the vote for each voter. For our purposes, we will assume that the attacker cannot corrupt the talliers in this way—that is, that the anonymity of the ballot is maintained by the tallier. An analogy for this assumption is that votes contained in envelopes are encrypted and the opening of the envelopes preserves anonymity through a mixnet (where the outer envelopes are discarded and unable to be tied to the inner envelope). We term this the *honestly processed envelope hypothesis*.

This hypothesis holds provided that anyone can observe the opening and processing of VBM ballots in person and detect if the talliers were tying ballots to voters or throwing ballots into the trash. Furthermore, most jurisdictions require a party balance in their elections offices, such that two workers of opposing parties must complete tasks like opening and counting ballots together (see, e.g., [236]). This limits the possibility of one party seeking an advantage by cheating, and we take this practice as an assumption to our model as well. Note that these assumptions do not preclude the adversary for corrupting the talliers and producing a fraudulent election outcome.

On the other end of the voting channel, standard VBM schemes typically allow for the filling out of the ballot in the clerk's office, and therefore VBM in theory satisfies the assumption that voters can cast their ballots out of sight of the coercer. We term this the *private voting hypothesis*.

In practice, this assumption may not hold, as in the case where the coercer can prevent the voter from going to a public location or if the voter cannot reasonably travel to the clerk's office due to distance or barriers to mobility. Therefore, we cannot assume that VBM satisfied the private voting hypothesis.

Most VBM schemes typically only allow each voter to "cast" their vote once: once the ballot is in the mail, the voter cannot do anything to cast another ballot or cancel out their vote. We term this the *unique ballot submission hypothesis*.

7.2.4 Examining VBM

JCJ requires that a voting scheme have three properties: correctness, verifiability, and coercion-resistance.

7.2.4.1 Correctness

Adopting the JCJ definition of correctness with our modified VBM protocols, we find that VBM does prevent double-voting (a voter submitting more than one ballot). However, the attacker has a method of defeating the election's correctness: ballot forging. Wet-ink signatures, of the type used in VBM to authenticate ballots, are a very poor form of authentication. A significant amount of discussion of the ease with which these signatures can be forged exists in forensics literature [39, 114]. Signature verification is a known imperfect process, with numerous incidents occurring over the years that have unduly disenfranchised voters or otherwise hindered election process [94, 163, 223]. Furthermore, signature accept and reject rates vary widely even within the same state [209]. An attacker can take advantage of these weakness to mount a **ballot forging attack**, which allows them to submit a fraudulent ballot by forging the voter's signature. This could cause a ballot from a voter not under the adversary's control to appear in the final tally. As our verify function would find that the reported outcome agrees with the paper trail and $\mathscr{B}\mathscr{B}_{\text{VBM}}$, a ballot forging attack could defeat the correctness of our election.

7.2.4.2 Verifiability

VBM under the JCJ definition of verifiability is not verifiable, as an attacker can craft an election in which the reported tally of the election may not be exactly identical to the actual tally computed on the paper ballots. However, as verify relies on risk-limiting audits to provide verifiability, it is not possible that the attacker could craft a tally in which the outcome reflected by the paper ballots differs from the reported result without detection and correction. Therefore, we adopt the definition of verifiability provided by RLAs:

Definition 11. An election is **verifiable** if the adversary cannot forge a tally with a different outcome than the tally computed over $\mathscr{B}$.

Because RLAs preclude the ability of the attacker to forge an election result that is not statistically supported by the set of paper ballots, VBM achieves this definition of verifiability.

7.2.4.3 Coercion-resistance

As VBM does not preserve an anonymous channel in which to cast a ballot, it cannot provide coercion-resistance. An attacker can perform the following attacks under our model to win the coercion-resistance experiment specified in [125]:

- **Ballot harvesting**—an attacker can force a voter to sign their envelope and surrender their unmarked ballot, enabling the attacker to vote in place of the voter. This allows an attacker to mount a simulation attack.
- **Ballot seizure**—an attacker can force the voter to surrender their voted, sealed ballot, mounting a forced-abstention attack.
- **Blackmail**—an attacker can threaten or pay the voter to abstain, and verify they have done so by checking $\mathscr{B}\mathscr{B}_{\text{VBM}}$.

7.2.5 Realistic Coercion Mitigation

We note that while coercion of any voter should not be permitted by any voting system, it is often impossible to achieve coercion-resistance in real-world voting systems. Therefore, we propose a weaker definition, **coercion-hardness**.

Definition 12. A voting scheme is **coercion-hard** if the adversary must corrupt a significant fraction of voters F to alter the outcome of the election.

Assuming a real-world adversary, the degree of tampering required to defeat this definition is directly tied to the margin of the election. Formally, if $c_{\mathscr{A}}$ is the coercer's preferred choice that would not win without coercion, m is the true margin in an election in votes (the difference between the votes for the least-vote-getting winner $c_{\mathscr{W}}$ and $c_{\mathscr{A}}$, if there was no coercion), a coercer would need to change at least $F = \frac{m}{2}$ votes for $c_{\mathscr{W}}$ to $c_{\mathscr{A}}$ to change the outcome. This can be done if the coercer can identify at least $\frac{m}{2}$ voters who would otherwise vote for $c_{\mathscr{W}}$ and submit forged ballots for them before they can vote, or if they can force abstention by at least m voters who would vote for $c_{\mathscr{W}}$ and $\sum_{i=1}^{N} m_i$ voters who voted for other losing candidates that got more votes than $c_{\mathscr{A}}$, where m_i is the margin between those candidates and $c_{\mathscr{A}}$ respectively. In a forced-abstention attack, $F = m + \sum_{i=1}^{N} m_i$. We propose that a reasonable definition of F is $F \equiv m$, such that the attacker must corrupt at least m voters to sway the election outcome.

In the worst case, the attacker would prefer the runner up, the candidate with the most votes that did not win. This would minimize F. However, except in unusually close or small elections, F is likely to be at least hundreds of votes, if not thousands or more. This means that an adversary who has not corrupted the talliers will have to do significant amount of work in a coordinated fashion to violate the correctness property. VBM's reliance on paper ballots and one-time voting makes widespread coercion arduous, if not flat out infeasible. Moreover, VBM is not usually the only means of voting available to voters, and an attacker wishing to change the result of the election by only attacking VBM will have to do *even more* coercion of VBM voters to compensate for votes they cannot coerce in the non-VBM modes of voting. For our purposes, we will assume that VBM is the only mode of voting, and that the attacker's candidate is the most vote-getting loser for worst-case analysis.

Because the adversary in VBM can perform simulation attacks, VBM is not coercion hard. We propose a mechanisms to prevent corruption of the talliers and simulation attacks in Section 7.3, proposing a VBM scheme that is correct, verifiable, and coercion-hard.

7.3 Improving VBM

It is clear that vote-by-mail as implemented in the U.S. is a fairly insecure system. Not only does it permit coercion, but it also cannot guarantee correctness. These failings stem from several places: the weak, wet-ink signature-based authentication model, a lack of individual verifiability, and a lack of an anonymous voting channel. In this section, we present modifications to VBM as described above that can dramatically improve these issues.

7.3.1 Improving Authentication

In order to improve VBM authentication, we propose implementing HMAC-based one-time-pads (HOTPs) [235] to replace voters' wet-ink signatures on their ballot envelopes. A shared secret between the voter and the registrars is established during the registration phase, and can be derived from data that the registrar gets from the voter (above, the i passed into register). The key can be downloaded by the voter from the voter registration website during the registration phase, or it can be made available offline by the registrar via mail or in-person transaction.

The key can then be used by the voter and the registrar to generate HOTPs. For the registrar, new HOTPs can be generated in the election management software each time a ballot is mailed to the voter, and stored until the voter mails their ballot back. The voter can generate a new HOTP each time they receive a ballot to vote using an app provided by the registrar.

Implementing HOTPs inherits their strong authentication properties, significantly improving the security of VBM. Combining authentication with HOTPs and trustworthy policies concerning election officials, VBM can satisfy the correctness property we specified in Section 7.2.4.1, as well as the one specified in [125]. Because an attacker can no longer forge a ballot for a voter not under their control, and ballots cannot be destroyed once received by the election officials, it is infeasible for the attacker to affect the tally by simulating a voter not under their control.

7.3.2 Improving the Voting Channel

Now that we can guarantee that our election scheme is correct, we turn our attention to coercion. A major issue with VBM as it stands is that voters only have one opportunity to submit their ballot, and once their ballot is submitted (and authenticated) the vote is counted. Some states, like Michigan, allow voters to submit their ballots multiple times, cancelling out previously received ballots. Implementing this policy dramatically improves voters' access to an anonymous channel, and significantly weakens the coercer's ability to determine if a coerced voter voted in the way they wanted.

Even if re-voting cannot be permitted, our HOTP scheme can be used to revoke a submitted ballot, which in the worst case reduces all attacks to forced abstention attacks. As we discuss above, if the adversary cannot change the coerced voter's vote, but can only discard it, it at least doubles the amount of work they need to do to change the election outcome. To revoke a submitted ballot, a voter would simply have to iterate their HOTP to a different value than was written on the envelope. This functionality can be supported on the election website, and could require multiple HOTP codes to be generated for authentication to prevent spurious iterations. When the ballot is received, because the HOTP value will not longer match what is on the ballot, the ballot can be discarded.

Since we are still only allowing one vote attempt per voter in this scheme, a revocation need not cause $\mathscr{B}\mathscr{B}_{\text{VBM}}$ to reset for a voter. This way, an attacker can also no longer tell if their vote was counted, further weakening their ability to coerce voters.

7.3.3 VBM Improved

Now that we have identified several areas where VBM can be improved, we present VBM-Improved, which implements our suggested changes. First, we need to modify our existing functions to include our improvements. We introduce H to correspond to a keyed-HMAC, where the first positional argument corresponds to the key and the second to the data to be hashed. The functions tally and verify remain the same as before.

- **Registering**: The function $\mathsf{register}(SK_{\mathscr{R}}, i, \boldsymbol{L}) \rightarrow \{\perp, (K_i, C_i)\}$ takes the registrar's secret key and the voter's information i, and returns a shared secret key K_i and counter C_i if the registration was successful, and adds the voter to the roll $\boldsymbol{L}$. Otherwise, the registration returns nothing.

- **Voting**: The function $\mathsf{vote}(i, H(K_i, C_i), PK_{\mathscr{T}}, n_{\mathscr{C}}, \beta) \rightarrow b$ takes in the voter's information i, the HMAC-based one-time-password computed over the shared secret K and counter C, tallier's public key, the slate of candidates, and the voter's choices and produces a sealed paper ballot b for tallying.

- **Authentication**: The function $\mathsf{authenticate}(i, H(K_i, C_i), b, \boldsymbol{L}, \mathscr{B}\mathscr{B}_{\text{VBM}}) \rightarrow \{\perp, b_{open}\}$ takes in the voter's information i, the one-time-password based on the voter's key K_i and counter C_i, a sealed ballot b, the list of eligible voters $\boldsymbol{L}$ and $\mathscr{B}\mathscr{B}_{\text{VBM}}$. It computes $H(K_{\mathscr{R}_i}, C_{\mathscr{R}_i})$ using the key and counter held by the registrar corresponding to the voter in $\boldsymbol{L}$, and if it matches, then the ballot is opened and added to the set of ballots $\mathscr{B}$ for tallying, and $\mathscr{B}\mathscr{B}_{\text{VBM}}$ is updated to `True` for the voter. Otherwise, the ballot is discarded.

Next, we need to introduce two new functions, revoke and revote, which will allow voters to either revoke their already-cast ballot, or to vote a new one.

- **Revoking**: The function $\mathsf{revoke}(H(K_i, C_i)) \to \perp$ takes in a one-time-password and the voter's information and removes the corresponding ballot, if it exists, from the set of ballots to be tallied.

- **Re-voting**: The function $\mathsf{revote}(H(K_i, C_i), PK_{\mathscr{T}}, n_{\mathscr{C}}, \beta) \to b$ takes in a new one-time-password derived from the voter's shared secret and new ballot. This function removes any other ballots corresponding to this voter from the set to be tallied, and then includes the new ballot submitted by this function.

With these new constraints of our authenticate function, it is no longer possible for the attacker to add or remove ballots from voters not under their control. While we assume that $\mathscr{R}$ and $\mathscr{T}$ are honest, systems on which they rely, like ballot scanners, may not be. However, verify will correct incorrect outcomes produced by malicious scanner software and the like. Therefore, with this set of assumptions VBM-Improved satisfies the correctness definition set out in [125] and our modified definition of verifiability from above.

A non-trivial risk with any HOTP-based system is leakage of the shared secret. Such a leakage would result in an attacker gaining the ability to forge ballots for voters outside of their control, reducing the benefits of HOTPs back to written signatures. However, provided that best-practices laid out in [235] are followed by the election officials and the HOTP scheme is implemented securely, we believe this risk to be fairly minimal.

Another issue with the HOTP scheme is if an attacker can force a voter to surrender their device or access to the HOTP codes, the attacker could in principle forge votes for this voter. However, this is essentially the same as the attacker controlling the voter, and does not violate the correctness property.

The addition of revote and revoke also strengthen the anonymity of the vote-casting channel. Even though we can still not assume that one exists, voters now have more options for behaving in ways that can defeat the coercer. Because $\mathscr{BB}_{\text{VBM}}$ is not updated when a ballot is revoked or re-voted, the adversary no longer can tell whether a voter has voted the ballot they chose. The adversary can still perform forced-abstention attacks, but their capability to sway the election outcome in this way has been reduced by at least half, as forced abstention attacks require at least twice as much coercion as simulation attacks. Because of this, VBM-Improved achieves the property of coercion-hardness, in addition to being verifiable and correct.

Figure 7.2 shows a comparison between VBM and VBM-Improved.

7.4 Discussion

In this section we discuss some properties of our scheme, suggest where it can be applied, and discuss some future work.

7.4.1 Related Work

A significant amount of work that has been done on coercion resistance has focused on Internet voting schemes, and solutions typically involve fairly heavy-weight cryptographic solutions which also require specialized hardware [134]. Because the United States has no unified identification card, unlike, for example, Estonia [211], solutions that rely on national identification do not work. Moreover, no individual state or jurisdiction has smart-card enabled IDs, so providing the public key infrastructure required by most existing coercion-resistant schemes like those in [17, 64, 71, 72, 125, 136, 145, 178, 199] is likely not feasible.

Analyses of the security of a few vote-by-mail schemes do exist [130, 189], however they do not examine VBM schemes through the lens of coercion resistance as canonically defined.

Although the coercion guarantees provided by VBM-Improved may not be as strong as those provided in many systems in the existing literature, we believe its light weight and compatibility

VBM Voting Scheme

1. **Setup:** The candidate slate $\mathscr{C}$ for the election is published by the registrars with appropriate integrity protection.
2. **Registration:** The identities and eligibility of would-be participants in the election are verified by $\mathscr{R}$. Given successful verification, an individual becomes a registered voter, $\mathscr{R}$ stores their signature, and mails them a ballot. Previously registered voters may not need to re-register or request a ballot. $\mathscr{R}$ publishes a voter roll $\boldsymbol{L}$ and initialized $\mathscr{B}\mathscr{B}_{\text{VBM}}$.
3. **Voting:** Referring to the candidate slate $\mathscr{C}$, registered voters use their credentials to cast ballots. They place their ballots into a sealed envelope, sign the envelop with the same signature that $\mathscr{R}$ has on file, and mail the ballot back.
4. **Authenticating:** Once $\mathscr{R}$ receives the ballot, they compare the signature with the one on file. If it matches, the ballot is opened and included in the tally, and flip that voter's bit to true on $\mathscr{B}\mathscr{B}_{\text{VBM}}$. Otherwise, they may communicate to the voter that the signature does not match.
5. **Tallying:** The authority $\mathscr{T}$ processes the contents of the the set of ballots so as to produce a tally vector $\boldsymbol{X}$ specifying the outcome of the election.
6. **Verification:** Any player, whether or not a participant in the election, can query $\mathscr{B}\mathscr{B}_{\text{VBM}}$ to ascertain if a voter's ballot was received, as well as summing $\mathscr{B}\mathscr{B}_{\text{VBM}}$ to ensure that the number of voters is greater than or equal to the sum of $\boldsymbol{X}$. A post-election audit is performed on the ballots in a public ceremony to confirm the outcome.

VBM-Improved Voting Scheme

1. **Setup:** The candidate slate $\mathscr{C}$ for the election is published by the registrars with appropriate integrity protection.
2. **Registration:** The identities and eligibility of would-be participants in the election are verified by $\mathscr{R}$. Given successful verification, an individual becomes a registered voter, $\mathscr{R}$ generates a key K_i and counter C_i, and mails them a ballot. Previously registered voters may not need to re-register or request a ballot. $\mathscr{R}$ publishes a voter roll $\boldsymbol{L}$ and initialized $\mathscr{B}\mathscr{B}_{\text{VBM}}$.
3. **Voting:** Referring to the candidate slate $\mathscr{C}$, registered voters use their credentials to cast ballots. They place their ballots into a sealed envelope, generate a one-time-password using $H(K_i,C_i)$ and write it on their envelope, and mail the ballot back.
4. **Re-voting and revoking** If a voter wishes to vote again or to revoke the ballot they have already submitted, they invoke revote or revoke respectively. They can do this until the election ends.
5. **Authenticating:** Once $\mathscr{R}$ receives the ballot, they compute $H(K_{\mathscr{R}_i},C_{\mathscr{R}_i})$ and ensure that the value matches the one written on the ballot. If it matches, the ballot is opened and included in the tally, and flip that voter's bit to true on $\mathscr{B}\mathscr{B}_{\text{VBM}}$. Otherwise, they may communicate to the voter that the value does not match.
6. **Tallying:** The authority $\mathscr{T}$ processes the contents of the the set of ballots so as to produce a tally vector $\boldsymbol{X}$ specifying the outcome of the election.
7. **Verification:** Any player, whether or not a participant in the election, can query $\mathscr{B}\mathscr{B}_{\text{VBM}}$ to ascertain if a voter's ballot was received, as well as summing $\mathscr{B}\mathscr{B}_{\text{VBM}}$ to ensure that the number of voters is greater than or equal to the sum of $\boldsymbol{X}$. A post-election audit is performed on the ballots in a public ceremony to confirm the outcome.

Figure 7.2: **Comparison between VBM procedure and VBM-Improved procedure**

with existing schemes more than make up for this. Furthermore, it is not as if there are no guarantees of coercion prevention: the scheme does require significantly more effort on the part of the coercer to affect an election outcome.

7.4.2 Usability

As VBM-Improved relies on existing technical infrastructure, namely HOTPs, it inherits their usability properties, for better or for worse. HOTPs are most commonly deployed in two-factor authentication settings, and much work has been done about the user experience and usability of these systems. Several studies have examined a multitude of 2FA apps which use one-time passwords as their authentication mechanism [7, 76, 135, 188, 243, 244], in general noting wide ranges of attitudes and abilities to successfully use one-time-passwords.

Unlike the context of existing usability work on one-time passwords, those in VBM-improved must be correctly transcribed onto the ballot envelope. This may present another usability issue: human transcription of many-digit numbers is not well studied.[3]. If this does present an issue, other kinds of signature codes may be generated. Following after some of the usable password work [47], signatures codes may be random sentences constructed based on the output of the signature code function. This provides a neat property: even if the voter makes a mistake, it is unlikely that the mistake will make their signature code unverifiable: a typo in one word of a sentence does not obscure the content of teh sentence.

Further work is needed to establish what impact our scheme has on the usability of absentee voting. Better understanding how accurate humans are at transcribing numbers is paramount to making our scheme work. Moreover, requiring voters to perform an additional step in the process voting may have negative affects on voters' ability to vote successfully, and this also needs further study. However, we note that VBM-Improved is completely backwards compatible with existing VBM scheme, so voters who do not wish to use HOTPs or who have difficulty doing so can always default back to their established behavior patterns when voting by mail.

[3] [144] provides some discussion of handwriting recognition, that's somewhat unrelated. There has also been some work on CAPTCHAs, which is a similar task [54]

7.5 Conclusion

In this chapter we examined how even voting systems that deploy best-practices can still fall short in terms of security. We presented a remedy to the problem of absentee ballot signature validation. Signature validation will only become a more pressing problem as absentee voting expands in the United States and elsewhere. This work is a step towards solving voter authentication for remote voting, and we hope that future work will illuminate better mechanisms to solve this problem.

CHAPTER VIII

Conclusion

In this work, I have presented an argument that simply knowing how to improve election security is not enough. If security solutions are not adapted to specific local environments, if they do not form part of a fabric of security solutions covering the whole of an election system, or if there are even seemingly insignificant mistakes in their deployment, they may not provide enough security to defeat a motivated adversary. This last idea is worth emphasizing: we have seen motivated attackers present in the election space for the past few years, attempting to subvert election systems through a variety of means. Because of this, it is no longer sufficient to merely say "you should do X" when asked about election security. The devil is in the details.

As we have seen with voter-verified paper, slight deviations in the means of deployment can have disastrous effects. An attacker could hack software designed to support auditing images of paper ballots or the machines that produce those paper ballots. These attacks are not limited to the devices and systems I have discussed here; attacks on ballot printers in hand-marked solutions, on machines that facilitate postal voting or remote voting otherwise, or on software designed to facilitate risk-limiting audits are also possible and not well studied.

Risk-limiting audits present several challenges to their feasibility in many jurisdictions. By and large, the lack of expertise in the underlying statistics of RLAs and the software the supports them has proven a significant hindrance to their use. Fitting an RLA to a specific jurisdiction's needs is critical, as some jurisdictions simply cannot perform some kinds of audits because of technical or legal constraints. Even meeting these places where they are introduces drawbacks: while it is possible to conduct a perfectly parallelized ballot-polling audit on election night without a ballot manifest, does the cost to transparency and the risk of tired poll workers making mistakes outweigh the benefits?

One need only to look at the state of Virginia for an example of how an attempt to cope with the local environment and still run an "RLA" can fail spectacularly. Virginia's RLA law requires that jurisdictions be drawn from a hat and perform an RLA once every five years. The RLA must be performed after certification and cannot have an impact on the outcome of the election that is audited. Furthermore, no election in Virginia is wholly contained within one jurisdiction, so unless all jurisdictions containing an election are drawn at once (which is higly improbable), no audit could be valid for any race. While I maintain that it is important for technologies be adapted to the needs of the places using them, it is not always unreasonable to require that some places adapt to their technologies. In either case, the nuance of both the place and the technology must be understood for things to work.

Finally, even in systems that are widely regarded as doing everything right, high levels of security are not readily attained. The state of Colorado is frequently held up as the pinnacle of election security in the United States, as it relies almost exclusively on hand-marked, vote-by-mail ballots and performs ballot-comparison RLAs, the most efficient kind. But as we have seen in the last chapter, even this system can be subject to attack in a variety of ways. While I only addressed wet-ink signature forging, it is not hard to imagine a hack of a signature verification machine, or of the voter registration database that could put the entire election system in chaos overnight.

Nevertheless, none of these things exists in a vacuum. Since the 2016 election a wide variety of other security technologies not discussed here have been deployed in the U.S. End-point monitoring, intrusion detection systems, encrypted email, multifactor authentication, and many other technologies have been deployed in addition to an increased use of voter-verifiable paper and post-election audits to provide greater election security. To my knowledge, there is not much work in the academic literature examining how these technologies perform in the election setting, nor how they interplay with other technologies to improve election security.

8.1 Future Work

While I have mentioned a few areas for future work above, here I set aside some space for specific work and questions that I have in mind.

More work on verifiability. While more and more work is being done on the topic of what makes a voting system verifiable, much more work is needed. A lot of the current discussion focuses exclusively on ballot marking devices, while hand-marked paper is assumed to be verifiable. This is not always so; one can imagine an attack on the printer that produces the paper ballots wherein some small fraction of paper ballots is misprinted so that it will not scan correctly, or that two candidates' names are flipped so that voters will think they are voting for one person but actually vote for the other. This type of attack is also possible on a BMD, of course, but there is not much evidence about hand-marked paper. Even a replication of the Everett et al. study examining error rates with hand-marked paper would do a great deal to provide vital evidence to the discussion [95].

Risk-limiting audits. As auditing pilots have picked up across the U.S. and the world, a good deal of data has been produced about which auditing practices work and which don't. Work is needed that reckons with these experiences, as well as new audit practices that build on the knowledge gained. For example, my impression from having participated in dozens of audits at this point is that workload is not as big a deal as once thought. Other factors, like how the sample sizes degenerate as margins get small, or how to provide likelihood of finishing in one round seem to be the prevailing concerns with election officials. Work that takes this into account would be useful for future auditing technology.

Coercion and remote voting. As the COVID-19 pandemic continues, more and more jurisdictions are seeking to broaden the set of voters that uses vote-by-mail or even online voting. The actual rates of coercion in remote voting is not well understood, and will indeed be a difficult thing to measure. However, it ought to be possible to do so, following best practices established in fields like the study of stalkerware and intimate partner violence.

Other election systems. Russia penetrated two voter registration systems in 2016. How secure are these systems? What about election night reporting? How likely is an attack where state ENR systems are taken out and replaced with fakes?

Complex system externalities. Many of the systems that elections rely on are much more complex than just voting. One only has to read the software load out on a voting system to see that it isn't just marking and counting, but formatting and mailing and testing and so on. How likely is it that some obscure piece of software present in a voting system provides a point of entry for malware? Postal voting relies on a communication channel that is not well understood in terms of computer security: the post office. Most mail interfaces with dozens of computers in the course of its traversal, but how well secured are those systems? What potential attacks could be carried out on the post office?

8.2 Looking Ahead

So there is a great deal of work to do. The ideas I have presented here merely scratch the surface of what is possible and what we have seen in terms of attacks on elections the world over. Moreover, I have largely disregarded the political and sociological dimensions of examining these technologies. Integrity, verifiability, coercion, correctness, etc. are all malleable, and the choices of their definitions and how they are modelled are necessarily political. It is all well and good to say that a voting system has eligibility verifiability, but this completely belies the moral and ethical questions about who should be allowed to vote in the first place. It is all well and good to employ technologies like hand-marked paper that are thought to provide software-independence and cast-as-intended verifiability, but what does it matter if this necessarily excludes vulnerable populations like voters who have difficulty marking paper by hand or do not have an address to which a vote-by-mail ballot can be delivered? It is all well and good to decry the insecurity of voting by mail and the potential for fraud that it creates, but if the alternative is forcing voters to risk their lives and show up to vote during a disease outbreak, is defending against an exceptionally rare threat really that important?

The overall point is that while this protracted document has supported the notion that election security is difficult and involves significantly more nuance and care than obvious at first blush, even that is somehow an understatement. There is a tremendous amount of future work needed in the technical space to explore the oft-wielded claims about the security of hand-marked paper, risk-limiting audits, etc., but also work about disinformation, how voting systems impact psychology and sociology, how secure systems can embed equity into their operation, and so on.

My hope is that the approach I have taken here can influence work to those ends. Security is not something achieved, it is something improved. There is no magic bullet; there is no black-and-white in this space. In order to truly make gains and improve security in elections, it is absolutely crucial to understand the finer details of the solutions that are suggested and the places to which they can be applied. It is exactly these details that make election security harder than you think.

Appendices

Appendix A

Voter-verified Paper

1.1 Poll Worker Script

Our poll workers followed four versions of the script below: a baseline version, and three variants that each add one line.

VARIANT 1: Before the voter begins using the BMD, a poll worker asks them to check their ballot before it is scanned.

VARIANT 2: Before the voter deposits the ballot, a poll worker informs them that it is the official record of the vote.

VARIANT 3: Before the voter deposits the ballot, a poll worker asks whether they have carefully reviewed each selection.

When Subject Arrives (POLL WORKER A)

*Hello! Before you begin, please fill out this Institutional Review Board consent form. [*Point to form and pen.*] If you have any questions, feel free to ask.*

You are about to participate in a study about the usability of a new type of voting machine. You will be using one of these voting machines to make selections on your ballot, which will be a truncated version of the Ann Arbor 2018 midterm ballot. Once you are finished, your ballot will be printed *from the printer beneath the machine, and you can review your ballot and deposit it in the ballot box over there. [*Point out ballot box.*] Feel free to vote your political preference or not; no identifying information will be collected that could match you with your votes. If you would like to quit at any time during the study, just say so.*

Variant 1*: Please remember to check your ballot carefully before depositing it into the scanner.*

You may begin at any time.

Before Subject Deposits Ballot (Poll Worker B)

Variant 2*: Please keep in mind that the paper ballot is the official record of your vote.*

Variant 3*: Have you carefully reviewed each selection on your printed ballot?*

After Subject Deposits Ballot (Poll Worker B)

*Thank you for participating! You are now finished with the study, and should fill out the exit survey. [*Point to debrief survey computers.*]*

After Subject Completes Exit Survey (Poll Worker B)

*Thank you for your participation! You are now finished. If you have any questions about this study, you may ask them now, although I am unable to answer some questions due to the nature of the research. Here is a debrief form. [*Hand subject a debrief form.*] If you think of anything after you leave, you can reach [me/the principle investigators] through the information on the debrief form.*

If you know anyone who might like to participate, please refer them here; we will be here [remaining time].

Thank you again for participating!

1.2 Recruitment Script

An investigator used the following script to recruit library patrons to participate in the study:

Hello, do you have 10 minutes to participate in a study about a new kind of voting machine that *is used in elections across the United States? This study will consist of voting using our voting machine and depositing a printed paper ballot into a ballot box, and then filling out a survey about the experience. If you would like to participate, we will need you to first sign a consent form. We will provide a flyer at the end of your participation with information about the study. We cannot make all details available at this time, but full details and research results will be made available within six months of the conclusion of this study. We thank you for your consideration and hope you choose to participate!*

1.3 Slate of Candidates for Directed Voting Condition

We randomly generated a slate of candidates and provided a printed copy to voters in certain experiments. The handout voters received is reproduced below:

Race	Candidate(s)
Governor and Lieutenant Governor	Bill Gelineau and Angelique Chaiser Thomas
Secretary of State	Mary Treder Lang
Attorney General	Lisa Lane Gioia
United States Senator	Debbie Stabenow
Representative in Congress 12th District	Jeff Jones
Member of State Board of Education **(Vote for 2)**	Tiffany Tilley Mary Anne Hering
Regent of the University of MIchigan **(Vote for 2)**	Jordan Acker Joe Sanger
Trustee of Michigan State University **(Vote for 2)**	Mike Miller Bruce Campbell
Justice of the Supreme Court **(Vote for 2)**	Megan Kathleen Cavanagh Kerry Lee Morgan
Judge of Court of Appeals 3rd District Incumbent Position **(Vote for 2)**	Jane Marie Beckering Douglas B. Shapiro
Judge of Circuit Court 22nd Circuit Incumbent Position **(Vote for 2)**	Timothy Patrick Connors Carol Kuhnke
Judge of Probate Court Incumbent Position	Darlene A. O’Brien
Judge of District Court 14A District Incumbent Position	Thomas B. Bourque

BIBLIOGRAPHY

[1] A review of robust post-election audits. Technical report. Accessed July 20 2020.

[2] Claudia Z Acemyan, Philip Kortum, Michael D Byrne, and Dan S Wallach. Usability of voter verifiable, end-to-end voting systems: Baseline data for Helios, Prêt à Voter, and Scantegrity II. *USENIX Journal of Election Technology and Systems (JETS)*, 2:26–56, 2014.

[3] Claudia Z Acemyan, Philip Kortum, Michael D Byrne, and Dan S Wallach. From error to error: Why voters could not cast a ballot and verify their vote with Helios, Prêt à Voter, and Scantegrity II. *USENIX Journal of Election Technology and Systems (JETS)*, 3:1–25, 2015.

[4] Claudia Ziegler Acemyan and Philip Kortum. Does the polling station environment matter? the relation between voting machine layouts within polling stations and anticipated system usability. In *Proceedings of the Human Factors and Ergonomics Society*, volume 59, pages 1066–1070, 2015.

[5] Claudia Ziegler Acemyan, Philip Kortum, Michael D Byrne, and Dan S Wallach. Summative usability assessments of STAR-Vote: A cryptographically secure e2e voting system that has been empirically proven to be easy to use. *Human Factors*, 2018.

[6] Claudia Ziegler Acemyan, Philip Kortum, and David Payne. Do voters really fail to detect changes to their ballots? An investigation of ballot type on voter error detection. *Proceedings of the Human Factors and Ergonomics Society*, 57:1405–1409, 2013.

[7] Claudia Ziegler Acemyan, Philip Kortum, Jeffrey Xiong, and Dan S Wallach. 2fa might be secure, but it's not usable: A summative usability assessment of google's two-factor authentication (2fa) methods. In *Proceedings of the Human Factors and Ergonomics Society*

Annual Meeting, volume 62, pages 1141–1145. SAGE Publications Sage CA: Los Angeles, CA, 2018.

[8] Ben Adida and Ron L. Rivest. Scratch and Vote: Self-contained paper-based cryptographic voting. In *ACM Workshop on Privacy in the Electronic Society*, WPES '06, pages 29–40, 2006.

[9] Devdatta Akhawe and Adrienne Porter Felt. Alice in Warningland: A large-scale field study of browser security warning effectiveness. In *22nd USENIX Security Symposium*, pages 257–272, 2013.

[10] Hazim Almuhimedi, Florian Schaub, Norman Sadeh, Idris Adjerid, Alessandro Acquisti, Joshua Gluck, Lorrie Faith Cranor, and Yuvraj Agarwal. Your location has been shared 5,398 times! A field study on mobile app privacy nudging. In *33rd ACM Conference on Human Factors in Computing Systems*, CHI, pages 787–796, 2015.

[11] Joël Alwen, Rafail Ostrovsky, Hong-Sheng Zhou, and Vassilis Zikas. Incoercible multi-party computation and universally composable receipt-free voting. In *Advances in Cryptology—CRYPTO 2015*, pages 763–780. Springer, 2015.

[12] Ann Arbor District Library. Classic shop drop! Plus, fish election results!, August 2019. https://aadl.org/node/396262.

[13] Ann Arbor District Library. Mock the vote, July 2019. https://aadl.org/node/395686.

[14] Ann Arbor District Library. Mock voting @ AADL, September 2019. https://aadl.org/node/397364.

[15] Andrew W Appel, Richard A DeMillo, and Philip B Stark. Ballot-marking devices cannot ensure the will of the voters. *Election Law Journal: Rules, Politics, and Policy*, 2020.

[16] Diego F Aranha and Jeroen van de Graaf. The good, the bad, and the ugly: Two decades of e-voting in brazil. *IEEE Security & Privacy*, 16(6):22–30, 2018.

[17] Roberto Araújo, Sébastien Foulle, and Jacques Traoré. A practical and secure coercion-resistant scheme for remote elections. In *Dagstuhl Seminar Proceedings*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2008.

[18] Ariel J. Feldman and J. Alex Halderman and Edward W. Felten. Security analysis of the Diebold AccuVote-TS voting machine. In *USENIX/ACCURATE Electronic Voting Technology Workshop*, EVT '07, August 2007.

[19] J.A. Aslam, R.A. Popa, and R.L. Rivest. On auditing elections when precincts have different sizes. In *2008 USENIX/ACCURATE Electronic Voting Technology Workshop, San Jose, CA, 28–29 July*, 2008.

[20] Andrea Bajcsy, Ya-Shian Li-Baboud, and Mary Brady. Systematic measurement of marginal mark types on voting ballots. Technical report, National Institute for Standards and Technology, 2015.

[21] Belhassen Bayar and Matthew C Stamm. A deep learning approach to universal image manipulation detection using a new convolutional layer. In *Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security*, pages 5–10. ACM, 2016.

[22] Sevinç Bayram, İsmail Avcıbaş, Bülent Sankur, and Nasir Memon. Image manipulation detection with binary similarity measures. In *2005 13th European Signal Processing Conference*, pages 1–4. IEEE, 2005.

[23] Sevinç Bayram, Ismail Avcibas, Bülent Sankur, and Nasir D Memon. Image manipulation detection. *Journal of Electronic Imaging*, 15(4):041102, 2006.

[24] Benjamin B Bederson, Bongshin Lee, Robert M Sherman, Paul S Herrnson, and Richard G Niemi. Electronic voting system usability issues. In *21st ACM Conference on Human Factors in Computing Systems*, CHI, pages 145–152, 2003.

[25] Karen Brinson Bell. Numbered memo 2020-06: Voter instructions for voting systems. https://s3.amazonaws.com/dl.ncsbe.gov/sboe/numbermemo/2020/Numbered%20Memo%202020-06_Voter%20Instructions%20for%20Voting%20Systems.pdf, 2020. Accessed 20 July 2020.

[26] Susan Bell, Josh Benaloh, Michael D. Byrne, Dana DeBeauvoir, Bryce Eakin, Gail Fisher, Philip Kortum, Neal McBurnett, Julian Montoya, Michelle Parker, Olivier Pereira, Philip B. Stark, Dan S. Wallach, and Michael Winn. STAR-vote: A secure, transparent, auditable, and

reliable voting system. *USENIX Journal of Election Technology and Systems*, 1(1), August 2013.

[27] M. Grant Belton, Philip Kortum, and Claudia Z. Acemyan. How hard can it be to place a ballot into a ballot box? Usability of ballot boxes in tamper resistant voting systems. *Journal of Usability Studies*, 10(4):129–139, 2015.

[28] Josh Benaloh. Ballot casting assurance via voter-initiated poll station auditing. In *USENIX/ACCURATE Electronic Voting Technology Workshop*, EVT '07, August 2007.

[29] Josh Benaloh. Administrative and public verifiability: Can we have both? In *USENIX/ACCURATE Electronic Voting Technology Workshop*, EVT '08, August 2008.

[30] Josh Benaloh, Douglas Jones, Eric Lazarus, Mark Lindeman, and Philip B. Stark. Soba: Secrecy-preserving observable ballot-level audit. In *proc. Proc. USENIXAccurate Electronic Voting Technology Workshop*, 2011.

[31] Josh Benaloh, Ronald Rivest, Peter YA Ryan, Philip Stark, Vanessa Teague, and Poorvi Vora. End-to-end verifiability, 2015. arXiv:1504.03778.

[32] Josh Benaloh and Dwight Tuinstra. Receipt-free secret-ballot elections. In *26th ACM Symposium on Theory of Computing*, STOC '94, pages 544–553, 1994.

[33] Josh Daniel Cohen Benaloh. *Verifiable Secret-ballot Elections*. PhD thesis, Yale, 1987. AAI8809191.

[34] David Bernhard, Véronique Cortier, David Galindo, Olivier Pereira, and Bogdan Warinschi. Sok: A comprehensive analysis of game-based ballot privacy definitions. In *2015 IEEE Symposium on Security and Privacy*, pages 499–516. IEEE, 2015.

[35] Matthew Bernhard, Josh Benaloh, J Alex Halderman, Ronald L Rivest, Peter YA Ryan, Philip B Stark, Vanessa Teague, Poorvi L Vora, and Dan S Wallach. Public evidence from secret ballots. In *2nd International Joint Conference on Electronic Voting*, E-Vote-ID, pages 84–109, 2017.

[36] Matthew Bernhard, Kartikeya Kandula, Jeremy Wink, and J Alex Halderman. Unclearballot:

Automated ballot image manipulation. In *International Joint Conference on Electronic Voting*, pages 14–31. Springer, 2019.

[37] Matthew Bernhard, Allison McDonald, Henry Meng, Jensen Hwa, Nakul Bajaj, Kevin Chang, and J Alex Halderman. Can voters detect malicious manipulation of ballot marking devices? In *41st IEEE Symposium on Security and Privacy*, 2020.

[38] Robert Bernstein, Anita Chadha, and Robert Montjoy. Overreporting voting: Why it happens and why it matters. *Public Opinion Quarterly*, 65(1):22–44, 2001.

[39] Carolyne Bird, Bryan Found, Kaye Ballantyne, and Doug Rogers. Forensic handwriting examiners' opinions on the process of production of disguised and simulated signatures. *Forensic Science International*, 195(1-3):103–107, 2010.

[40] M. Blaze, J. Braun, H.Hursti, J. Lorenzo Hall, M. MacAlpine, and J. Moss. DEFCON 25 voting village report, September 2017. https://www.defcon.org/images/defcon-25/DEF%20CON%2025%20voting%20village%20report.pdf.

[41] M. Blaze, J. Braun, H. Hursti, D. Jefferson, M. MacAlpine, and J. Moss. DEFCON 26 voting village report, September 2018. https://www.defcon.org/images/defcon-26/DEF%20CON%2026%20voting%20village%20report.pdf.

[42] Alan Blinder. Inside a fly-by-night operation to harvest ballots in north carolina. https://www.nytimes.com/2019/02/20/us/north-carolina-voter-fraud.html, February 2019.

[43] Michelle Blom, Andrew Conway, Dan King, Laurent Sandrolini, Philip B Stark, Peter J Stuckey, and Vanessa Teague. You can do rlas for irv. *arXiv preprint arXiv:2004.00235*, 2020.

[44] Michelle Blom, Peter J Stuckey, and Vanessa Teague. Raire: Risk-limiting audits for irv elections. In *4th International Joint Conference on Electronic Voting*, E-Vote-ID, 2019.

[45] Michelle Blom, Peter J Stuckey, Vanessa J Teague, and Ron Tidhar. Efficient computation of exact IRV margins, 2015. arXiv:1508.04885.

[46] Greg Bluestein. Georgia Democrats set new primary turnout record, outpacing GOP vot-

ers. https://www.ajc.com/blog/politics/georgia-democrats-set-new-primary-turnout-record-outpacing-gop-voters/fotxE4Udba0e0q6QvDBZ8M/, 2020.

[47] Joseph Bonneau, Cormac Herley, Paul C Van Oorschot, and Frank Stajano. The quest to replace passwords: A framework for comparative evaluation of web authentication schemes. In *2012 IEEE Symposium on Security and Privacy*, pages 553–567. IEEE, 2012.

[48] D. Bowen et al. Top-to-bottom review of voting machines certified for use in California. Technical report, California Secretary of State, 2007. https://www.sos.ca.gov/elections/ovsta/frequently-requested-information/top-bottom-review/.

[49] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[50] Steven Brams. *Mathematics and democracy*. Princeton Univ. Press, 2008.

[51] C. Bravo-Lillo, L. F. Cranor, J. Downs, and S. Komanduri. Bridging the gap in computer security warnings: A mental model approach. *IEEE Security & Privacy*, 9(2):18–26, March 2011.

[52] Cristian Bravo-Lillo, Saranga Komanduri, Lorrie Faith Cranor, Robert W. Reeder, Manya Sleeper, Julie Downs, and Stuart Schechter. Your attention please: Designing security-decision UIs to make genuine risks harder to ignore. In *9th Symposium on Usable Privacy and Security*, SOUPS, 2013.

[53] US Census Bureau. Voting Districts, 1994.

[54] Elie Bursztein, Steven Bethard, Celine Fabry, John C Mitchell, and Dan Jurafsky. How good are humans at solving CAPTCHAs? A large scale evaluation. In *Security and Privacy (SP), 2010 IEEE Symposium on*, pages 399–413. IEEE, 2010.

[55] Michael D. Byrne and Susan Bovair. A working memory model of a common procedural error. *Cognitive Science*, 21(1):31–61, 1997.

[56] Michael D Byrne, Kristen K Greene, and Sarah P Everett. Usability of voting systems: baseline data for paper, punch cards, and lever machines. In *25th ACM Conference on Human Factors in Computing Systems*, CHI, pages 171–180, 2007.

[57] Joseph A. Calandrino, J. Alex Halderman, and Edward W. Felten. Machine-assisted election

auditing. In *USENIX/ACCURATE Electronic Voting Technology Workshop*, EVT '07, August 2007.

[58] Bryan A Campbell and Michael D Byrne. Now do voters notice review screen anomalies? a look at voting system usability. In *USENIX Electronic Voting Technology Workshop/Workshop on Trustworthy Elections*, EVT/WOTE, 2009.

[59] Ran Canetti and Rosario Gennaro. Incoercible multiparty computation. In *37th IEEE Symposium on Foundations of Computer Science*, FOCS '96, pages 504–513, 1996.

[60] Richard Carback, David Chaum, Jeremy Clark, John Conway, Aleksander Essex, Paul S. Herrnson, Travis Mayberry, Stefan Popoveniuc, Ronald L. Rivest, Emily Shen, Alan T. Sherman, and Poorvi L. Vora. Scantegrity II municipal election at Takoma Park: The first E2E binding governmental election with ballot privacy. In *18th USENIX Security Symposium*, August 2010.

[61] Anthony Cardillo, Nicholas Akinyokun, and Aleksander Essex. Online voting in ontario municipal elections: A conflict of legal principles and technology? In *International Joint Conference on Electronic Voting*, pages 67–82. Springer, 2019.

[62] Carter Center. Expert study mission report—Internet voting pilot: Norway's 2013 parliamentary elections, March 2014. http://www.regjeringen.no/upload/KRD/Kampanjer/valgportal/valgobservatorer/2013/Rapport_Cartersenteret2013.pdf.

[63] David Cary. Estimating the margin of victory for instant-runoff voting. In *USENIX/ACCURATE Electronic Voting Technology Workshop / Workshop on Trustworthy Elections*, EVT/WOTE '11, August 2011.

[64] Pyrros Chaidos, Véronique Cortier, Georg Fuchsbauer, and David Galindo. Beleniosrf: A non-interactive receipt-free electronic voting scheme. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1614–1625, 2016.

[65] D. Chaum. SureVote: Technical overview. In *IAVoSS Workshop on Trustworthy Elections*, WOTE '01, 2001.

[66] 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.

[67] Stephen Checkoway, Ariel J. Feldman, Brian Kantor, J. Alex Halderman, Edward W. Felten, and Hovav Shacham. Can DREs provide long-lasting security? The case of return-oriented programming and the AVC Advantage. In *USENIX Electronic Voting Technology / Workshop on Trustworthy Elections*, EVT/WOTE '09, August 2009.

[68] Berj Chilingirian, Zara Perumal, Ronald L Rivest, Grahame Bowland, Andrew Conway, Philip B Stark, Michelle Blom, Chris Culnane, and Vanessa Teague. Auditing australian senate ballots. *arXiv preprint arXiv:1610.00127*, 2016.

[69] Kevin Kwong-tai Chung, Victor Jun Dong, and Xiaoming Shi. Electronic voting method for optically scanned ballot, July 18 2006. US Patent 7,077,313.

[70] November 6, 2018 general election. https://dochub.clackamas.us/documents/drupal/f4e7f0fb-250a-4992-918d-26c5f726de3c.

[71] Jeremy Clark and Urs Hengartner. Selections: Internet voting with over-the-shoulder coercion-resistance. In *International Conference on Financial Cryptography and Data Security*, pages 47–61. Springer, 2011.

[72] Michael R Clarkson, Stephen Chong, and Andrew C Myers. Civitas: Toward a secure voting system. In *2008 IEEE Symposium on Security and Privacy (sp 2008)*, pages 354–368. IEEE, 2008.

[73] Clear Ballot. ClearAudit. https://clearballot.com/products/clear-audit.

[74] Clear Ballot Group. ClearAccess administrators guide, 2015. https://www.sos.state.co.us/pubs/elections/VotingSystems/systemsDocumentation/ClearBallot/ClearAccess/ClearAccessAdministratorsGuideRev4-0-r0.pdf.

[75] W.G. Cochran. *Sampling Techniques: 3d Ed.* Wiley, 1977.

[76] Jessica Colnago, Summer Devlin, Maggie Oates, Chelse Swoopes, Lujo Bauer, Lorrie Cranor, and Nicolas Christin. "it's not actually that horrible" exploring adoption of two-

factor authentication at a university. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 1–11, 2018.

[77] Colorado Secretary of State. Audit Center. https://www.sos.state.co.us/pubs/elections/auditCenter.html.

[78] Frederick G Conrad, Benjamin B Bederson, Brian Lewis, Emilia Peytcheva, Michael W Traugott, Michael J Hanmer, Paul S Herrnson, and Richard G Niemi. Electronic voting eliminates hanging chads but introduces new usability challenges. *International Journal of Human-Computer Studies*, 67(1):111–124, 2009.

[79] A. Cordero, D. Wagner, and D. Dill. The role of dice in election audits – extended abstract. In *IAVoSS Workshop On Trustworthy Elections (WOTE 2006)*, 2006.

[80] Andrea Cordova, Liz Howard, and Lawrence Norden. Voting machine security: Where we stand a few months before the New Hampshire primary. Brennan Center, 2019. https://www.brennancenter.org/analysis/voting-machine-security-where-we-stand-six-months-new-hampshire-primary.

[81] Véronique Cortier, David Galindo, Ralf Küsters, Johannes Mueller, and Tomasz Truderung. Sok: Verifiability notions for e-voting protocols. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 779–798. IEEE, 2016.

[82] Lorrie Faith Cranor. A framework for reasoning about the human in the loop. In *1st Conference on Usability, Psychology, and Security*, UPSEC. USENIX, 2008.

[83] Cybersecurity and Infrastructure Security Agency. Security tip (st18-006) website security. https://www.us-cert.gov/ncas/tips/ST18-006, 2020 (accessed 2020-6-15).

[84] Menno De Jong, Joris Van Hoof, and Jordy Gosselt. Voters' perceptions of voting technology: Paper ballots versus voting machine with and without paper audit trail. *Social Science Computer Review*, 26(4):399–410, 2008.

[85] Stéphanie Delaune, Steve Kremer, and Mark Ryan. Verifying privacy-type properties of electronic voting protocols: A taster. In *Towards Trustworthy Elections*, pages 289–309. Springer, 2010.

[86] Richard DeMillo, Robert Kadel, and Marilyn Marks. What voters are asked to verify affects ballot verification: A quantitative analysis of voters' memories of their ballots, 2018. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3292208.

[87] Verena Distler, Marie-Laure Zollinger, Carine Lallemand, Peter B. Roenne, Peter Y. A. Ryan, and Vincent Koenig. Security—visible, yet unseen? In *37th ACM Conference on Human Factors in Computing Systems*, CHI, 2019.

[88] Washington County Elections Division. Vote-by-mail made easy. https://www.co.washington.or.us/assessmenttaxation/elections/votebymail/index.cfm.

[89] Dominion Voting. Auditmark. https://www.dominionvoting.com/pdf/DD%20Digital%20Ballot%20AuditMark.pdf.

[90] Dominion Voting. Cambridge Case Study. https://www.dominionvoting.com/field/cambridge, 2014.

[91] Paul Egan. Number of Democratic ballots cast in Michigan's presidential primary sets record. https://www.freep.com/story/news/politics/elections/2020/03/11/michigan-presidential-primary-high-voter-turnout/5019250002/, 2020.

[92] Serge Egelman, Lorrie Faith Cranor, and Jason Hong. You've been warned: An empirical study of the effectiveness of web browser phishing warnings. In *26th ACM Conference on Human Factors in Computing Systems*, CHI, pages 1065–1074, 2008.

[93] Election Integrity Oregon. https://www.electionintegrityoregon.org.

[94] Tyler Estep. Voting, civil rights groups home in on gwinnett's absentee rejections. https://www.ajc.com/news/local-govt--politics/just-group-sues-kemp-gwinnett-elections-board-over-ballot-rejections/1qMxof9sA0um6w32vmrG2I/, October 2018.

[95] Sarah P Everett, Michael D Byrne, and Kristen K Greene. Measuring the usability of paper ballots: efficiency, effectiveness, and satisfaction. In *Proceedings of the Human Factors and Ergonomics Society*, volume 50, pages 2547–2551, 2006.

[96] Sarah P Everett, Kristen K Greene, Michael D Byrne, Dan S Wallach, Kyle Derr, Daniel Sandler, and Ted Torous. Electronic voting machines versus traditional methods: improved

preference, similar performance. In *26th ACM Conference on Human Factors in Computing Systems*, CHI, pages 883–892, 2008.

[97] Hany Farid. Digital forensics in a post-truth age. *Forensic science international*, 289:268–269, 2018.

[98] Ezra Fieser. People Openly Sell Votes for $20 in the Dominican Republic. http://www.bloomberg.com/news/articles/2016-05-16/people-openly-sell-their-votes-for-20-in-the-dominican-republic, May 2016.

[99] Mike Fitts. SC chooses new voting machines that will print paper ballots but some fear it's not safe. The Post and Courier, June 10, 2019. https://www.postandcourier.com/article_f86632ce-8b83-11e9-8dab-5fb7858906cc.html.

[100] Bryan Found, J Sita, and D Rogers. The development of a program for characterizing forensic handwriting examiners' expertise: Signature examination pilot study. *Journal of Forensic Document Examination*, 12:69–80, 1999.

[101] Stephen Fowler. Georgia awards new voting machine contract to Dominion Voting Systems. Georgia Public Broadcasting, July 29, 2019. https://www.gpbnews.org/post/georgia-awards-new-voting-machine-contract-dominion-voting-systems.

[102] Daniel E. Gaines. Preserving the private vote? state adopts new policy on accessible ballots. https://www.marylandmatters.org/2019/06/28/preserving-the-private-vote-state-adopts-new-policy-on-accessible-ballots/. Accessed 20 July 2020.

[103] Kristian Gjøsteen. The Norwegian Internet voting protocol. In *3rd International Conference on E-Voting and Identity*, VoteID '11, 2011.

[104] Stephen N Goggin and Michael D Byrne. An examination of the auditability of voter verified paper audit trail (VVPAT) ballots. In *USENIX Electronic Voting Technology Workshop*, EVT, 2007.

[105] Rop Gonggrijp and Willem-Jan Hengeveld. Studying the nedap/groenendaal es3b voting computer: A computer security perspective. In *Proceedings of the USENIX workshop on accurate electronic voting technology*, pages 1–1. USENIX Association, 2007.

[106] Kristen K Greene, Michael D Byrne, and Sarah P Everett. A comparison of usability between voting methods. In *USENIX Electronic Voting Technology Workshop*, EVT, 2006.

[107] Gurchetan S Grewal, Mark D Ryan, Sergiu Bursuc, and Peter YA Ryan. Caveat coercitor: Coercion-evidence in electronic voting. In *34th IEEE Symposium on Security and Privacy*, pages 367–381, 2013.

[108] Rhode Island Risk Limiting Audit Working Group. Pilot implementation study of risk-limiting audit methods in the state of rhode island. The Rhode Island Risk-Limiting Audit Working Group (of which I am a member) conducted pilot audits in January 2019.

[109] Rolf Haenni and Oliver Spycher. Secure internet voting on limited devices with anonymized dsa public keys. *EVT/WOTE*, 11, 2011.

[110] Thomas Haines, Sarah Jamie Lewis, Olivier Pereira, and Vanessa Teague. How not to prove your election outcome. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 784–800, 2019.

[111] J Alex Halderman and Vanessa Teague. The New South Wales iVote system: Security failures and verification flaws in a live online election. In *5th International Conference on E-Voting and Identity*, VoteID '15, August 2015.

[112] J.L. Hall, L.W. Miratrix, P.B. Stark, M. Briones, E. Ginnold, F. Oakley, M. Peaden, G. Pellerin, T. Stanionis, and T. Webber. Implementing risk-limiting post-election audits in California. In *2009 Workshop on Electronic Voting Technology/Workshop on Trustworthy Elections*, pages 19–19. USENIX Association, 2009.

[113] Sue Halpern. Can our ballots be both secret and secure? https://www.newyorker.com/news/the-future-of-democracy/can-our-ballots-be-both-secret-and-secure, 2020.

[114] Heidi H Harralson. *Developments in handwriting and signature identification in the digital age*. Routledge, 2014.

[115] Lane A Hemaspaandra, Rahman Lavaee, and Curtis Menton. Schulze and ranked-pairs voting are fixed-parameter tractable to bribe, manipulate, and control. In *International Conference on Autonomous Agents and Multiagent Systems*, pages 1345–1346, 2013.

[116] Paul S Herrnson, Benjamin B Bederson, Bongshin Lee, Peter L Francia, Robert M Sherman, Frederick G Conrad, Michael Traugott, and Richard G Niemi. Early appraisals of electronic voting. *Social Science Computer Review*, 23(3):274–292, 2005.

[117] Paul S Herrnson, Richard G Niemi, Michael J Hanmer, Benjamin B Bederson, Fred Conrad, and Michael Traugott. The not so simple act of voting: An examination of voter errors with electronic voting. In *Annual Meeting of the Southern Political Science Association*, 2006.

[118] Paul S Herrnson, Richard G Niemi, Michael J Hanmer, Benjamin B Bederson, Frederick G Conrad, and Michael Traugott. The importance of usability testing of voting systems. In *USENIX Electronic Voting Technology Workshop*, EVT, 2006.

[119] Paul S Herrnson, Richard G Niemi, Michael J Hanmer, Peter L Francia, Benjamin B Bederson, Frederick G Conrad, and Michael W Traugott. Voters' evaluations of electronic voting systems: Results from a usability field study. *American Politics Research*, 36(4):580–611, 2008.

[120] Indiana Fiscal Policy Institute. Vote centers and election costs: A study of the fiscal impact of vote centers in Indiana, 2010. https://www.in.gov/sos/elections/files/IFPI_Vote_Centers_and_Election_Costs_Report.pdf.

[121] Theron Ji, Eric Kim, Raji Srikantan, Alan Tsai, Arel Cordero, and David A Wagner. An analysis of write-in marks on optical scan ballots. In *EVT/WOTE*, 2011.

[122] Mark, verify, scan. https://jocoelection.org/Mark-Verify-Scan. Accessed July 20 2020.

[123] Douglas Jones and Barbara Simons. *Broken ballots: Will your vote count?* CSLI Publications, 2012.

[124] Douglas W Jones. On optical mark-sense scanning. In *Towards Trustworthy Elections*, pages 175–190. Springer, 2010.

[125] A. Juels, D. Catalano, and M. Jakobsson. Coercion-resistant Electronic Elections. In *ACM Workshop on Privacy in the Electronic Society*, WPES '05, pages 61–70, November 2005.

[126] Tyler Kaczmarek, John Wittrock, Richard Carback, Alex Florescu, Jan Rubio, Noel Runyan, Poorvi L. Vora, and Filip Zagórski. Dispute resolution in accessible voting systems: The

design and use of audiotegrity. In *E-Voting and Identify - 4th International Conference, Vote-ID 2013, Guildford, UK, July 17-19, 2013. Proceedings*, pages 127–141, 2013.

[127] Diana Kasdan. Early voting: What works. https://www.brennancenter.org/sites/default/files/publications/VotingReport_Web.pdf.

[128] Aggelos Kiayias and Moti Yung. Self-tallying elections and perfect ballot secrecy. In *5th International Workshop on Practice and Theory in Public Key Cryptosystems*, PKC '02, pages 141–158, 2002.

[129] Aggelos Kiayias, Thomas Zacharias, and Bingsheng Zhang. End-to-end verifiable elections in the standard model. In *Advances in Cryptology—EUROCRYPT 2015*, pages 468–498. Springer, 2015.

[130] Christian Killer and Burkhard Stiller. The swiss postal voting process and its system and security analysis. In *International Joint Conference on Electronic Voting*, pages 134–149. Springer, 2019.

[131] Tadayoshi Kohno, Adam Stubblefield, Aviel D. Rubin, and Dan S. Wallach. Analysis of an electronic voting system. In *25th IEEE Symposium on Security and Privacy*, 2004.

[132] Philip Kortum, Michael D Byrne, and Julie Whitmore. Voter verification of bmd ballots is a two-part question: Can they? mostly, they can. do they? mostly, they don't. *arXiv preprint arXiv:2003.04997*, 2020.

[133] Steve Kremer, Mark Ryan, and Ben Smyth. Election verifiability in electronic voting protocols. In *4th Benelux Workshop on Information and System Security*, WISSEC '09, November 2009.

[134] Kristjan Krips and Jan Willemson. On practical aspects of coercion-resistant remote voting systems. In *International Joint Conference on Electronic Voting*, pages 216–232. Springer, 2019.

[135] Kat Krol, Eleni Philippou, Emiliano De Cristofaro, and M Angela Sasse. "they brought in the horrible key ring thing!" analysing the usability of two-factor authentication in uk online

banking. *Proceedings of the Workshop on Usable Secuirty and Privacy (USEC'15)*, February 2015.

[136] Oksana Kulyk. *Extending the Helios Internet Voting Scheme Towards New Election Settings*. PhD thesis, Technische Universität, 2017.

[137] Ralf Küsters, Tomasz Truderung, and Andreas Vogt. Accountability: Definition and relationship to verifiability. In *17th ACM Conference on Computer and Communications Security*, CCS '10, pages 526–535, 2010.

[138] Ralf Küsters, Tomasz Truderung, and Andreas Vogt. Verifiability, privacy, and coercion-resistance: New insights from a case study. In *32nd IEEE Symposium on Security and Privacy*, pages 538–553, 2011.

[139] Ralf Küsters, Tomasz Truderung, and Andreas Vogt. A game-based definition of coercion resistance and its applications. *Journal of Computer Security*, 20(6):709–764, 2012.

[140] M. Lindeman, M. Halvorson, P. Smith, L. Garland, V. Addona, and D. McCrea. Principles and best practices for post-election audits, September 2008.

[141] M. Lindeman, N. McBurnett, K. Ottoboni, and P.B. Stark. Next steps for the Colorado risk-limiting audit (CORLA) program, March 2018. https://arxiv.org/pdf/1803.00698.pdf.

[142] M. Lindeman, P. B. Stark, and V. S. Yates. BRAVO: Ballot-polling risk-limiting audits to verify outcomes. In *USENIX Electronic Voting Technology Workshop / Workshop on Trustworthy Elections*, EVT/WOTE '12, August 2012.

[143] M. Lindeman and P.B. Stark. A gentle introduction to risk-limiting audits. *IEEE Security & Privacy*, 10:42–49, 2012.

[144] Greg Little, Lydia B Chilton, Max Goldman, and Robert C Miller. Turkit: tools for iterative tasks on mechanical turk. In *Proceedings of the ACM SIGKDD workshop on human computation*, pages 29–30. ACM, 2009.

[145] Philipp Locher and Rolf Haenni. Receipt-free remote electronic elections with everlasting privacy. *Annals of Telecommunications*, 71(7-8):323–336, 2016.

[146] S. Lohr. *Sampling: design and analysis*. Nelson Education, 2009.

[147] Julie Mack. Who votes in Michigan? A demographic breakdown. MLive, 2018. https://www.mlive.com/news/erry-2018/11/340b0f9c406363/who-votes-in-michigan-a-demogr.html.

[148] Thomas R. Magrino, Ronald L. Rivest, Emily Shen, and David Wagner. Computing the margin of victory in IRV elections. In *USENIX Electronic Voting Technology Workshop / Workshop on Trustworthy Elections*, EVT/WOTE '11, 2011.

[149] Sharon Maneki and Ben Jackson. Re: Comments on Ballot Marking Devices usage for the 2018 elections, 2017. Letter to Maryland State Board of Elections, citing SBE data.

[150] Karola Marky, Oksana Kulyk, Karen Renaud, and Melanie Volkamer. What did I really vote for? On the usability of verifiable e-voting schemes. In *36th ACM Conference on Human Factors in Computing Systems*, CHI, 2018.

[151] Karola Marky, Oksana Kulyk, and Melanie Volkamer. Comparative usability evaluation of cast-as-intended verification approaches in Internet voting. In Hanno Langweg, Michael Meier, Bernhard C. Witt, and Delphine Reinhardt, editors, *SICHERHEIT 2018*, pages 197–208, Bonn, 2018. Gesellschaft für Informatik e.V.

[152] Maryland House of Delegates. House Bill 1278: An act concerning election law – postelection tabulation audit. http://mgaleg.maryland.gov/2018RS/bills/hb/hb1278E.pdf.

[153] Maryland State Board of Elections. 2016 post-election audit report. http://dlslibrary.state.md.us/publications/JCR/2016/2016_22-23.pdf, 12 2016.

[154] Maryland State Board of Elections. December 15, 2016 meeting minutes. https://elections.maryland.gov/pdf/minutes/2016_12.pdf, December 2016.

[155] P. McDaniel, M. Blaze, and G. Vigna. EVEREST: Evaluation and validation of election-related equipment, standards and testing. Technical report, Ohio Secretary of State, 2007. https://www.eac.gov/assets/1/28/EVEREST.pdf.

[156] W.R. Mebane and M. Bernhard. Voting technologies, recount methods and votes in Wisconsin and Michigan in 2016. *3rd Workshop on Advances in Secure Electronic Voting 2018*, 2018.

[157] Michigan Secretary of State. Secretary Benson announces post-election audit plans. https://www.michigan.gov/som/0,4669,7-192-26847-498858--,00.html.

[158] Tal Moran and Moni Naor. Receipt-free universally-verifiable voting with everlasting privacy. In *Advances in Cryptology—CRYPTO 2006*, pages 373–392. Springer, 2006.

[159] Sarah Morin, Grant McClearn, Neal McBurnett, Poorvi L Vora, and Filip Zagórski. A note on risk-limiting bayesian polling audits for two-candidate elections. 2020.

[160] National Academies of Sciences, Engineering, and Medicine. *Securing the Vote: Protecting American Democracy*. The National Academies Press, Washington, DC, 2018.

[161] National Conference of State Legislatures. Funding elections technology, 2019. https://www.ncsl.org/research/elections-and-campaigns/funding-election-technology.aspx.

[162] National Conference of State Legislatures. Post-election audits, January 2019. http://www.ncsl.org/research/elections-and-campaigns/post-election-audits635926066.aspx.

[163] Brian Naylor. Sign here: Why elections officials struggle to match voters' signatures. https://www.npr.org/2018/11/17/668381260/sign-here-why-elections-officials-struggle-to-match-voters-signatures?utm_campaign=storyshare&utm_source=twitter.com&utm_medium=social, November 2018.

[164] Richard G Niemi and Paul S Herrnson. Beyond the butterfly: The complexity of U.S. ballots. *Perspectives on Politics*, 1(2):317–326, 2003.

[165] Michigan Bureau of Elections. *Election Officials'manual*. 2019. Accessed 20 July 2020.

[166] California Secretary of State. Ballot status. https://www.sos.ca.gov/elections/ballot-status/, 2020 (accessed 2020-6-15).

[167] Colorado Secretary of State. Signature verification guide. https://www.sos.state.co.us/pubs/elections/docs/SignatureVerificationGuide.pdf.

[168] Colorado Secretary of State. Mail-in Ballot FAQs. https://www.sos.state.co.us/pubs/elections/FAQs/mailBallotsFAQ.html, 2020 (accessed 2020-6-15).

[169] Georgia Secretary of State. How to Check the Status of Your Absentee or Provisional Ballot. https://sos.ga.gov/index.php/general/how_to_check_the_status_of_your_absentee_by_mail_or_provisional_ballot, 2018 (accessed 2020-6-15).

[170] Michigan Secretary of State. Register to vote A step-by-step guide. https://www.michigan.gov/sos/0,4670,7-127-1633_8716_8726_47669---,00.html, 2020 (accessed 2020-6-15).

[171] National Conference of State Legislatures. All-mail elections (aka vote-by-mail). https://www.ncsl.org/research/elections-and-campaigns/all-mail-elections.aspx, 2020 (accessed 2020-6-15).

[172] National Conference of State Legislatures. Covid-19 and elections. https://www.ncsl.org/research/elections-and-campaigns/state-action-on-covid-19-and-elections.aspx, 2020 (accessed 2020-6-15).

[173] National Conference of State Legistlatures. Online Voter Registration. https://www.ncsl.org/research/elections-and-campaigns/electronic-or-online-voter-registration.aspx, 2020 (accessed 2020-6-15).

[174] M Maina Olembo and Melanie Volkamer. E-voting system usability: Lessons for interface design, user studies, and usability criteria. In *Human-Centered System Design for Electronic Governance*, pages 172–201. IGI Global, 2013.

[175] Kellie Ottoboni, Matthew Bernhard, J Alex Halderman, Ronald L Rivest, and Philip B Stark. Bernoulli ballot polling: a manifest improvement for risk-limiting audits. In *International Conference on Financial Cryptography and Data Security*, pages 226–241. Springer, 2019.

[176] Kellie Ottoboni, Philip B Stark, Mark Lindeman, and Neal McBurnett. Risk-limiting audits by stratified union-intersection tests of elections (SUITE). In *International Joint Conference on Electronic Voting*, pages 174–188. Springer, 2018.

[177] Panel on Nonstandard Mixtures of Distributions. *Statistical models and analysis in auditing: A study of statistical models and methods for analyzing nonstandard mixtures of distributions in auditing*. National Academy Press, Washington, D.C., 1988.

[178] Ştefan Patachi and Carsten Schürmann. Eos a universal verifiable and coercion resistant voting protocol. In *International Joint Conference on Electronic Voting*, pages 210–227. Springer, 2017.

[179] Sameer Patil, Roberto Hoyle, Roman Schlegel, Apu Kapadia, and Adam J. Lee. Interrupt

now or inform later? Comparing immediate and delayed privacy feedback. In *33rd ACM Conference on Human Factors in Computing Systems*, CHI, pages 1415–1418, 2015.

[180] Justin Petelka, Yixin Zou, and Florian Schaub. Put your warning where your link is: Improving and evaluating email phishing warnings. In *37th ACM Conference on Human Factors in Computing Systems*, CHI, 2019.

[181] Pew Charitable Trusts. Colorado voting reforms: Early results. https://www.pewtrusts.org/~/media/assets/2016/03/coloradovotingreformsearlyresults.pdf, 2016.

[182] Associated Press. Arizona gop sues to limit mail-in ballots in senate race. https://www.nbcnews.com/politics/elections/arizona-gop-sues-limit-mail-ballots-senate-race-n933866, November 2018.

[183] Pro V&V. Test report for EAC 2005 VVSG certification testing: Clear-Ballot Group ClearVote 1.4 voting system, 2017. https://www.eac.gov/file.aspx?A=kOBM5qPeI8KZlJyADXYTieiXLwsxw4gYKIVroEkEBMo%3D.

[184] W Quesenbery. Defining a summative usability test for voting systems. In *UPA Workshop on Voting and Usability*, 2004.

[185] Whitney Quesenbery. Ballot marking devices make voting universal. Center for Civic Design, 2019. https://civicdesign.org/ballot-marking-devices-make-voting-universal/.

[186] Whitney Quesenbery, John Cugini, Dana Chisnell, Bill Killam, and Ginny Reddish. Letter to the editor: Comments on “A methodology for testing voting systems”. *Journal of Usability Studies*, 2(2):96–98, 2007.

[187] Robert W. Reeder, Adrienne Porter Felt, Sunny Consolvo, Nathan Malkin, Christopher Thompson, and Serge Egelman. An experience sampling study of user reactions to browser warnings in the field. In *36th ACM Conference on Human Factors in Computing Systems*, CHI, 2018.

[188] Ken Reese, Trevor Smith, Jonathan Dutson, Jonathan Armknecht, Jacob Cameron, and Kent Seamons. A usability study of five two-factor authentication methods. In *Fifteenth Symposium on Usable Privacy and Security ({SOUPS} 2019)*, 2019.

[189] Andrew Regenscheid and Nelson Hastings. *A Threat Analysis on UOCAVA Voting Systems*. 2008.

[190] Rhode Island Board of Elections. Board of Elections conducts two additional successful post-election Risk Limiting Audits. https://www.ri.gov/press/view/37355.

[191] R. L. Rivest and E. Shen. A Bayesian method for auditing elections. In *USENIX Electronic Voting Technology Workshop / Workshop on Trustworthy Elections*, EVT/WOTE '12, August 2012.

[192] R.L. Rivest. ClipAudit: A simple risk-limiting post-election audit, 2017. https://arxiv.org/abs/1701.08312.

[193] Ronald L. Rivest. Consistent sampling with replacement. arXiv.

[194] Ronald L Rivest. On the notion of "software independence" in voting systems. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 366(1881):3759–3767, 2008.

[195] Ronald L Rivest. A sum of square roots (ssr) pseudorandom sampling method for election audits. *Massachusetts Institute of Technology*, 2008.

[196] Ronald L Rivest. DiffSum: A simple post-election risk-limiting audit. *CoRR abs/1509.00127*, 2015.

[197] Ronald L Rivest and Warren D Smith. Three voting protocols: ThreeBallot, VAV, and Twin. In *USENIX/ACCURATE Electronic Voting Technology Workshop*, EVT '07, August 2007.

[198] Peter Y. A. Ryan, David Bismark, James Heather, Steve Schneider, and Zhe Xia. Prêt à Voter: A voter-verifiable voting system. *IEEE Transactions on Information Forensics and Security*, 4(4):662–673, 2009.

[199] Peter YA Ryan, Peter B Rønne, and Vincenzo Iovino. Selene: Voting with transparent verifiability and coercion-mitigation. In *International Conference on Financial Cryptography and Data Security*, pages 176–192. Springer, 2016.

[200] Donald G Saari. *Geometry of voting*. Springer, 2012.

[201] C.-E. Särndal, B. Swensson, and J. Wretman. *Model assisted survey sampling*. Springer Science & Business Media, 2003.

[202] Anand D Sarwate, Stephen Checkoway, and Hovav Shacham. Risk-limiting audits and the margin of victory in nonplurality elections. *Statistics, Politics and Policy*, 4(1):29–64, 2013.

[203] ScannerOne. Kodak i5600. http://www.scannerone.com/product/KOD-i5600.html.

[204] Markus Schulze. A new monotonic, clone-independent, reversal symmetric, and condorcet-consistent single-winner election method. *Social Choice and Welfare*, 36(2):267–303, 2011.

[205] Ted Selker, Elizabeth Rosenzweig, and Anna Pandolfo. A methodology for testing voting systems. *Journal of Usability Studies*, 2(1):7–21, 2006.

[206] Ted Selker, Elizabeth Rosenzweig, and Anna Pandolfo. Reply to comment on: The Methodology for Testing Voting Systems by Whitney Quesenbery, John Cugini, Dana Chisnell, Bill Killam, and Ginny Redish. *Journal of Usability Studies*, 2(2):99–101, 2007.

[207] Runbeck Election Services. Runbeck automated signature verification. https://runbeck.net/election-solutions/ballot-software/automated-signature-verification/.

[208] Jodi Sita, Bryan Found, and Doug K Rogers. Forensic handwriting examiners' expertise for signature comparison. *Journal of Forensic Science*, 47(5):1–8, 2002.

[209] Daniel A. Smith. Vote-by-mail ballots cast in florida. https://www.aclufl.org/sites/default/files/aclufl_-_vote_by_mail_-_report.pdf, September 2018.

[210] Ben Smyth, Mark Ryan, Steve Kremer, and Mounira Kourjieh. Towards automatic analysis of election verifiability properties. In *Automated Reasoning for Security Protocol Analysis and Issues in the Theory of Security*, pages 146–163. Springer, 2010.

[211] Drew Springall, Travis Finkenauer, Zakir Durumeric, Jason Kitcat, Harri Hursti, Margaret MacAlpine, and J Alex Halderman. Security analysis of the Estonian Internet voting system. In *21st ACM Conference on Computer and Communications Security*, CCS '14, pages 703–715, 2014.

[212] Mayuri Sridhar and Ronald L Rivest. k-cut: A simple approximately-uniform method

for sampling ballots in post-election audits. In *International Conference on Financial Cryptography and Data Security*, pages 242–256. Springer, 2019.

[213] Matthew C Stamm and KJ Ray Liu. Forensic detection of image manipulation using statistical intrinsic fingerprints. *IEEE Transactions on Information Forensics and Security*, 5(3):492–506, 2010.

[214] P. B. Stark and D. A. Wagner. Evidence-based elections. *IEEE Security and Privacy Magazine*, 10(05):33–41, Sep.–Oct. 2012.

[215] P.B. Stark. Conservative statistical post-election audits. *Annals of Applied Statistics*, 2(2):550–581, 2008.

[216] Philip B Stark. Efficient post-election audits of multiple contests: 2009 california tests. In *CELS 2009 4Th annual conference on empirical legal studies paper*, 2009.

[217] Philip B. Stark. Super-simple Simultaneous Single-ballot Risk-limiting Audits. In *Proceedings of the 2010 International Conference on Electronic Voting Technology/Workshop on Trustworthy Elections*, EVT/WOTE'10, pages 1–16, Berkeley, CA, USA, 2010. USENIX Association.

[218] Philip B Stark. There is no reliable way to detect hacked ballot-marking devices, 2019. https://arxiv.org/abs/1908.08144.

[219] Philip B Stark. Sets of half-average nulls generate risk-limiting audits: Shangrla. 2020.

[220] Philip B Stark and Vanessa Teague. Verifiable European elections: Risk-limiting audits for d'hondt and its relatives. *USENIX Journal of Election Technology and Systems*, 3(1), 2014.

[221] Joshua Sunshine, Serge Egelman, Hazim Almuhimedi, Neha Atri, and Lorrie Faith Cranor. Crying wolf: An empirical study of SSL warning effectiveness. In *18th USENIX Security Symposium*, pages 399–416, 2009.

[222] M. Thompson. *Theory of sample surveys*, volume 74. CRC Press, 1997.

[223] Glenn Thrush, Audra D. S. Burch, and Frances Robles. In florida recount, sloppy signatures placed thousands of ballots in limbo. https://www.nytimes.com/2018/11/14/us/voting-signatures-matching-elections.html, November 2018.

[224] T Nicolaus Tideman. Independence of clones as a criterion for voting rules. *Social Choice and Welfare*, 4(3):185–206, 1987.

[225] Ian Traynor. Russia accused of unleashing cyberwar to disable Estonia, May 2007. http://www.theguardian.com/world/2007/may/17/topstories3.russia.

[226] Unisyn Voting Solutions. OpenElect OCS Auditor. https://unisynvoting.com/openelect-ocs/.

[227] United States Senate Select Committee on Intelligence. Report on russian active measures campaigns and interference in the 2016 U.S. election, 2019. https://www.intelligence.senate.gov/sites/default/files/documents/Report_Volume1.pdf.

[228] Dominique Unruh and Jörn Müller-Quade. Universally composable incoercibility. In *Advances in Cryptology–CRYPTO 2010*, pages 411–428. Springer, 2010.

[229] U.S. Census Bureau. QuickFacts: Ann Arbor, 2019. https://www.census.gov/quickfacts/annarborcitymichigan.

[230] U.S. Election Assistance Commission. Designing polling place materials. https://www.eac.gov/election-officials/designing-polling-place-materials/.

[231] U.S. Election Assistance Commission. Certificate of conformance: ClearVote 1.5. https://www.eac.gov/file.aspx?A=zgte4IhsHz%2bswC%2bW4LO6PxIVssxXBebhvZiSd5BGbbs%3d, 2019.

[232] U.S. Senate Select Committee on Intelligence. Russian targeting of election infrastructure during the 2016 election: Summary of initial findings and recommendations, May 2018. https://www.burr.senate.gov/imo/media/doc/RussRptInstlmt1-%20ElecSec%20Findings,Recs2.pdf.

[233] Verified Voting. Ballot marking devices. https://www.verifiedvoting.org/ballot-marking-devices/. Accessed 2019-09-30.

[234] Verified Voting Foundation. The Verifier: Polling place equipment, 2019. https://www.verifiedvoting.org/verifier/.

[235] Mountain View, David M'Raihi, Frank Hoornaert, David Naccache, Mihir Bellare, and Ohad

Ranen. HOTP: An HMAC-Based One-Time Password Algorithm. RFC 4226, December 2005.

[236] 24.2-115 Appointment, qualifications, and terms of officers of election. *Code of Virginia*, Title 24.2. Elections, 2016 (accessed 2020-15-6).

[237] Poorvi L Vora. Risk-limiting bayesian polling audits for two candidate elections. *arXiv preprint arXiv:1902.00999*, 2019.

[238] VSAP. Voting system for all people. https://vsap.lavote.net/.

[239] A. Wald. Sequential tests of statistical hypotheses. *Ann. Math. Stat.*, 16:117–186, 1945.

[240] Wall Street Journal. Election 2018: How we voted in the 2018 midterms, November 6, 2018. https://www.wsj.com/graphics/election-2018-votecast-poll/.

[241] Dan Wallach. Security and Reliability of Webb County's ES&S Voting System and the March 06 Primary Election. *Expert Report in Flores v. Lopez*, 2006.

[242] Dan S Wallach. On the security of ballot marking devices, 2019. https://arxiv.org/abs/1908.01897.

[243] Jake Weidman and Jens Grossklags. I like it, but i hate it: Employee perceptions towards an institutional transition to byod second-factor authentication. In *Proceedings of the 33rd Annual Computer Security Applications Conference*, pages 212–224, 2017.

[244] Catherine S Weir, Gary Douglas, Martin Carruthers, and Mervyn Jack. User perceptions of security, convenience and usability for ebanking authentication tokens. *Computers & Security*, 28(1-2):47–62, 2009.

[245] Michael S. Wogalter. Communication-human information processing (C-HIP) model. In Michael S. Wogalter, editor, *Handbook of Warnings*, chapter 5, pages 51–61. Lawrence Erlbaum Associates, Mahwah, NJ, 2006.

[246] Michael S. Wogalter and Kenneth R. Laughery. Warning! sign and label effectiveness. *Current Directions in Psychological Science*, 5(2):33–37, 1996.

[247] Scott Wolchok, Eric Wustrow, J. Alex Halderman, Hari K. Prasad, Arun Kankipati, Sai Krishna Sakhamuri, Vasavya Yagati, and Rop Gonggrijp. Security analysis of India's electronic

voting machines. In *17th ACM Conference on Computer and Communications Security*, CCS '10, October 2010.