

IN THE GENERAL COURT OF JUSTICE
SUPERIOR COURT DIVISION
20 CVS 5035

**AFFIDAVIT OF ANDREW W.
APPEL**

1. I am of legal age and competent to provide this affidavit. All the information herein is based on my own personal knowledge unless otherwise indicated.
2. My background, qualifications, and professional affiliations are set forth in my curriculum vitae, which is attached as Exhibit A. I have over 40 years' experience in computer science, and 15 years' experience studying voting machines and elections.
3. I am the Eugene Higgins Professor of Computer Science at Princeton University, where I have been on the faculty since 1986 and served as Department Chair from 2009-2015. I have also served as Director of Undergraduate Studies, Director of Graduate Studies, and Associate Chair in that department. I have served as Editor in Chief of ACM Transactions on Programming Languages and

Systems, the leading journal in my field. In 1998 I was elected a Fellow of the Association for Computing Machinery, the leading scientific and professional society in Computer Science.

4. I received an A.B. (1981) from Princeton University *summa cum laude* in Physics, and a PhD (1985) from Carnegie Mellon University in Computer Science.

5. I have taught undergraduate and graduate courses at Princeton University in programming, programming languages, software engineering, election machinery, software verification, and formal methods.

6. I have provided testimony on election technology before the U.S. House of Representatives (Subcommittee on Information Technology, 2016), the New Jersey legislature (several committees, on several occasions 2005-2018), the Superior Court of New Jersey (Mercer County, 2009; Cumberland County, 2011), the U.S. District Court for the Northern District of Georgia (2019), the New York State Board of Elections (2019), the Freeholders of Mercer County NJ (2017 and 2019) and Essex County NJ (2019).

7. I provided written testimony regarding electronic voting systems that was credited at length by the district court in *Curling v. Raffensperger*, 397 F. Supp. 3d 1334, 1354-56, 1366-67 (N.D. Ga. 2019). A declaration I authored in that case is appended hereto as Exhibit B.

8. I have published over 100 scientific articles and books, including many papers on computer security and several papers on voting machines, election technology, and election audits.

9. I have served as a peer-review referee for the Usenix Electronic Voting Technology workshop.

10. I have made a scientific study of the ExpressVote and other BMDs. In September 2018, I published a short article identifying a flaw in the ExpressVote.¹ Then, between November 2018 and March 2019, I conducted a research project with Professor Rich DeMillo of Georgia Tech and Professor

¹ Andrew Appel, *Serious design flaw in ESS ExpressVote touchscreen: "permission to cheat"*, Freedom-To-Tinker, September 14, 2018, <https://freedom-to-tinker.com/2018/09/14/serious-design-flaw-in-ess-expressvote-touchscreen-permission-to-cheat/>.

Philip Stark of U.C. Berkeley. That collaboration led to the publication of our joint paper, “Ballot Marking Devices (BMDs) cannot assure the will of the voters,” (by Appel/DeMillo/Stark) released in April 2019. This paper has since gone through a peer-review process and will appear in *Election Law Journal* later in 2020. Several findings from that paper are relevant to this lawsuit and, as such, it is appended to this declaration as Exhibit C.

11. I am not being compensated for my work in preparing this Declaration. I expect that my future expenses, if any, will be reimbursed.

12. In this affidavit I will explain: Any computerized voting machine, including the ExpressVote ballot marking device currently being used in North Carolina, can be “hacked” to make it cheat; our only feasible defense against computer hacking is to have paper ballots that the voters actually marked with their intended vote; such paper ballots can be counted by computer but may be recounted or audited by hand; *and that ballot cards marked by an ExpressVote voting machine are not reliably marked with the voters’ intended votes, and thus do not provide a substantial defense against hacking.*

THE EXPRESSVOTE – AND ALL OTHER ELECTRONIC VOTING MACHINES – CAN BE HACKED TO MAKE THEM CHEAT

13. All computer-based vote-recording and vote-counting machines can be “hacked” to make them cheat. That is, a person or persons can install fraudulent software that deliberately misrecords or miscounts votes, to alter the outcome of elections.

14. There are many ways to install fraudulent software in a computer—to “hack” it. Depending on the computer system, it may be possible to do it with physical access (replace a memory chip on the motherboard, or insert a cartridge or thumb-drive in a slot) or over a network. Modern computer systems have many layers of software, and an insecurity in any one of those layers can compromise the security of all the layers above it.² It is therefore implausible to say that any computer—or voting

² See pages 89-90 of: *Securing the Vote: Protecting American Democracy*, by National Academies of Science, Engineering, and Medicine (Lee C. Bollinger, Michael A. McRobbie, Andrew W. Appel, Josh

machine—is perfectly secure. As a practical matter, state and county officials cannot hope to make its computer systems perfectly secure against sophisticated attackers.

15. Some voting machines have no network connection, so it is sometimes claimed that they are “not connected to the Internet.” But every voting machine needs to be “told” before every election, what contests are on the ballot, and which candidates are running in those contests. This “Ballot Definition File” needs to be downloaded into every voting machine before every election. On voting machines with no direct network connection, this is done by installing a removable media (memory card, or thumb-drive) into the voting machine. But those memory cards must be “programmed” from some other computer, typically a county election management computer or private election contractor’s computer, that *is* sometimes connected to the Internet. It is well understood as a principle of computer security—and it has been demonstrated in practice on real voting machines—that fraudulent vote-stealing software can be made to propagate on those removable-media memory cards. Therefore, an attacker anywhere on the Internet could install fraudulent software on a county’s voting machines, even though those machines have no *direct* network connection.

16. No amount of “logic and accuracy testing” (LAT) can detect such hacking, because fraudulent software can easily be programmed to distinguish between LAT mode and real election mode.

HAND MARKED PAPER BALLOTS DEFEND AGAINST HACKING

17. For that reason, many countries avoid the use of computers to count votes: voters mark or select paper ballots by hand, and pollworkers count them. This works well in unitary parliamentary systems of government where, in a typical election, there is only one contest on the ballot. It does not work as well in the United States, which has a Federal system in which a single election may have many separate contests; vote-counting entirely by hand would be very time-consuming and could result in errors.

Benaloh, Karen Cook, Dana DeBeauvoir, Moon Duchin, Juan E. Gilbert, Susan L. Graham, Neal Kelley, Kevin J. Kennedy, Nathaniel Persily, Ronald L. Rivest, Charles Stewart III), <https://doi.org/10.17226/25120>, September 2018.

18. Most U.S. election jurisdictions (states, counties, or other jurisdictions), including many counties in North Carolina, use a system of optical-scan vote counting of hand-marked paper ballots to speed the tabulation process. This is the most secure system for feasibly conducting North Carolina elections and quickly tabulating results that I know of. Although the optical scanner is a computer, and thus could be hacked to make it cheat, the paper ballots marked by the voters can be recounted by human inspection, yielding the correct election outcome (identifying the true winner or winners, depending on the election) no matter what computers may have been hacked.

19. A full by-hand recount can detect and correct computer-based fraud (hacking), computer bugs and misprogramming, miscalibration of voting machines, or other mistakes. But full recounts are expensive and time-consuming. Methods of random audits, in which a small sample of the ballots are inspected, compared, and counted, can be much more efficient. A class of those methods called Risk-Limiting Audits (RLAs) can make strong statistical guarantees of effectiveness: any hack, bug, or miscalibration will be detected and corrected with high (and known) probability.

**BALLOT CARDS MARKED BY THE EXPRESSVOTE ARE NOT RELIABLY MARKED
WITH VOTERS' INTENDED VOTES AND THEREFORE CANNOT BE ADEQUATE
PROTECTED AGAINST HACKING**

20. I understand that Mecklenburg County and several other counties in North Carolina propose to have all voters, or all voters who use Vote Centers, mark their ballots using ExpressVote voting machines. The ExpressVote is a computerized ballot-marking device (BMD) made by Election Systems and Software (ES&S). I will explain the severe insecurities of ExpressVote BMDs that *cannot* be corrected by any kind of recount or random audit.

21. Based on my scientific study of the ExpressVote and other BMDs, I conclude that, like any computer-based voting machine, the ExpressVote can be “hacked.” That is, the ExpressVote’s vote-marking software can be replaced by fraudulent vote-stealing software that steals votes by recording

different votes on the paper ballot than what the voter indicated on the touchscreen.³ Logic and accuracy testing (LAT) cannot detect such fraud, because the software can easily be programmed to cheat only on the actual election day.⁴

22. The ExpressVote is insecure because (1) most voters do not inspect the printed-out paper ballot carefully enough to notice whether the ExpressVote has printed the same vote that they indicated on the touchscreen, and (2) even if some voters do notice, at most they can correct their own votes—they cannot prove the machine has been cheating—so their *neighbors* who did not carefully inspect their printed-out paper ballots will still have their votes stolen, and election outcomes can be successfully altered by hackers. The empirical evidence and consequent analysis supporting these conclusions have been described in a series of scientific papers.

23. DeMillo, Kadel, and Marks⁵ observed a real polling place in Tennessee, where voters used ExpressVotes to produce paper ballot cards, and then carried these ballot cards to an optical scanner. The researchers sat in a part of the room where pollwatchers were permitted—close enough to observe voters but not close enough to see which candidates the voters selected. The researchers observed that 47% of the voters did not look at the contents of the ballot card; and of the 53% that did look at the

³ Most voters indicate their votes on the ExpressVote using a touchscreen interface. The ExpressVote has another interface available to voters with visual impairments or other disabilities that render them unable to use the touchscreen: a device with buttons and audio. My analysis of the ExpressVote's insecurities applies regardless of which of these devices the voter uses; henceforth I will refer only to the touchscreen.

⁴ So-called “parallel testing” cannot reliably detect this fraud either; see: There is no Reliable Way to Detect Hacked Ballot-Marking Devices, by Philip B. Stark, August 21, 2019, <https://arxiv.org/abs/1908.08144>; and see Section 6 of the 2020 revision (to appear in *Election Law Journal*) of Appel, DeMillo, and Stark, Ballot-Marking Devices Cannot Assure the Will of the Voters, February 14, 2020, available at <https://www.cs.princeton.edu/~appel/papers/bmd-insecure.pdf>.

⁵ What Voters are Asked to Verify Affects Ballot Verification: A Quantitative Analysis of Voters' Memories of Their Ballots, by Richard DeMillo, Robert Kadel, and Marilyn Marks, (November 23, 2018). Available at SSRN: <https://ssrn.com/abstract=3292208>.

ballot, they spent an average of 3.9 seconds inspecting it. There were 18 contests on the ballot, so this is less than $\frac{1}{4}$ second per contest.

24. Bernhard et al.⁶ performed a controlled experiment: they set up BMDs in a public library in Michigan, and asked library patrons to participate in “a study about the usability of a new type of voting machine.” The BMDs were specially hacked to print, in one contest per paper ballot, a different candidate than the voter had selected. Only 7% of the voters reported the error to a poll worker, and only 8% reported the error on an exit survey.

25. The conclusion of both studies, and of earlier studies of “review screens” of touchscreen DREs, is that the vast majority of voters who use a touchscreen to indicate their ballot choices, do not carefully enough review their marked ballots to notice whether anything is marked differently than the vote they indicated on the screen.

26. I reviewed the Mecklenburg County Board of Elections’ instructional video about how to use its ExpressVote machines. The video can be found at <https://www.youtube.com/watch?v=9lPu-rTgoe8&feature=youtu.be>, and can be found via <https://www.mecknc.gov/BOE/voter/Pages/equipment.aspx>. The version of the video available as of the current date does not reference or instruct voters to review or even look at the printed ballot summary card before inserting it into the scanner.

27. If most voters don’t inspect their ExpressVote-marked ballots, then what are the consequences for the hackability (conversely, auditability) of elections? In our paper, we considered a scenario such as this one: an attacker wishes to change an election outcome from 53%-47% (a victory) to 48%-52% (a loss) for candidate A versus candidate B in some downballot race such as State Senator or Sheriff. To

⁶ Can Voters Detect Malicious Manipulation of Ballot Marking Devices? by Matthew Bernhard, Allison McDonald, Henry Meng, Jensen Hwa, Nakul Bajaj, Kevin Chang, and J. Alex Halderman. Accepted for publication, *IEEE Symposium on Security and Privacy*, May 2020.

do so, he programs the BMDs to alter 5% of the votes from A to B. Assuming only 10% of the voters inspect their ballots carefully in all the downballot races, then only 1 in 200 voters will notice.

28. If a voter notices that the paper ballot has a different candidate marked than they intended to vote for, the voter is supposed to inform a pollworker, who is then supposed to void that ballot and allow the voter to mark a fresh ballot. In this case (provided that the machine does not cheat again), the voter has corrected their vote. Consequently (because most voters won't notice), the machine succeeds in altering only 4.5% of the votes instead of 5% of the votes, and the reported outcome is 49%-51%, a loss for candidate A, instead of the true outcome 53%-47% (a win for candidate A) corresponding to what the voters indicated on the touchscreen.

29. You might think, "but some voters *caught the machine cheating red-handed*," in that they indicated candidate A on the touchscreen but found candidate B marked on the paper. But the voter cannot prove that the machine cheated: by the time the paper ballot is printed, the hacked software could alter what appears on the screen.

30. You might think, "if 1 in 200 voters reports that the machine is malfunctioning, that's strong evidence that the election has been hacked." But election officials cannot change an election outcome just because 0.5% of the voters report an error; if that were the practice, than small groups of voters could invalidate elections by fraudulently reporting that their ballots were misprinted.

31. You might think, "some sort of audit should catch any incorrect vote totals resulting from hacked ExpressVotes." But a recount or random audit can only check the tabulation of what's printed on the paper: it cannot go back in time and understand how that mark got made on the paper.

32. Therefore, ExpressVote-marked ballots are not meaningfully auditable or recountable: hacked ExpressVotes can manipulate results in a way that is almost certainly not going to be corrected in the real world.

33. In contrast, when a voter marks an optical-scan “bubble ballot” with a pen, no hackable computer intermediary stands between the voter’s *indication* of a vote (the mark made with the pen) and the *mark* that is read by human recounters or auditors. Hand-marked paper ballots are therefore auditable and recountable.

BAR CODES ON THE EXPRESSVOTE BALLOT CARD CAUSE FURTHER INSECURITY

34. The ExpressVote prints each selected choice in a contest in two forms: as a barcode (a pattern of vertical lines), and in human-readable block capital letters. Voters using the ExpressVote are expected to verify the human-readable printing on the paper ballot card, but the optical scanners that tabulate votes read only the barcodes.

35. **Barcodes are not human readable.** The whole purpose of a paper ballot is to be able to recount (or audit) the voters’ votes in a way independent of computers (which can be possibly hacked or buggy). If the official vote on the ballot card is the barcode, then it is impossible for voters to verify that the official vote they cast is the vote they expressed. Therefore, to even consider using BMDs that print barcodes (and I do not recommend doing so), recounts and audits in North Carolina should be based only on the human-readable portion of the paper ballot. Even so, audits based on untrustworthy paper trails suffer from the verifiability problems outlined above.


36. **Ballot cards with barcodes contain two different votes.** Suppose that recounts and audits in North Carolina are based on the human-readable portion of the paper ballot. Now, an ExpressVote-marked ballot card with both barcodes and human-readable text contains two different votes in each contest: the barcode (used for electronic tabulation), and the human-readable selection printout (official for audits and recounts). If an audit or recounted is conducted and this situation arises, it is not possible to determine whether the barcode or the human-readable text is “correct” insofar as it reflects the actual selection that the voter originally made on the ExpressVote touch-screen or keypad.

CONCLUSION

37. The ExpressVote, like all ballot-marking devices, prints ballots that do not necessarily record the vote expressed by the voter when they enter their selections on the touchscreen: hacking, bugs, and configuration errors can cause the ExpressVote to print votes that differ from what the voter entered (and verified on the screen). Such outcome-changing errors in BMD cannot be detected or corrected by any manual recount or audit of the ballot cards. Therefore, tabulations of ExpressVote ballots cannot be relied upon to express the will of the voters.

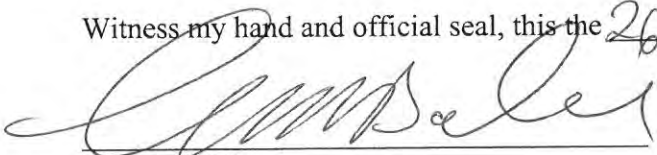
38. I declare under penalty of perjury that the foregoing is true and correct to the best of my knowledge.

This, the 26 day of June, 2020


Andrew W. Appel

I, the undersigned notary public, hereby certify that ANDREW W. APPEL personally appeared before me this day and acknowledged the due execution of this AFFIDAVIT.

Witness my hand and official seal, this the 26 day of June, 2020.


Notary Public

My commission expires 04/03, 2020

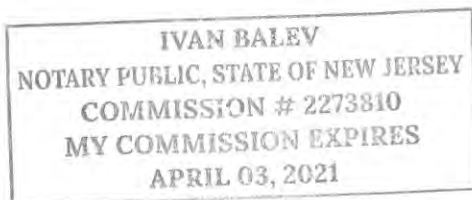


EXHIBIT A

Andrew W. Appel, Curriculum Vitae

Andrew W. Appel

Eugene Higgins Professor of Computer Science
Department of Computer Science, Princeton University
35 Olden Street, Princeton NJ 08540

appel@princeton.edu, +1-609-258-4627, fax: +1-609-258-2016
<https://www.cs.princeton.edu/~appel>

Research Interests

Software verification, programming languages, computer security, compilers, semantics, software engineering, information technology policy, elections and voting technology.

Education

A.B. *summa cum laude* (physics) Princeton University, 1981
Ph.D. (computer science) Carnegie-Mellon University, 1985

Professional Appointments

Princeton University, Princeton, NJ. Eugene Higgins Professor of Computer Science, since 2011; Department Chair, 2009-15; Professor of Computer Science, since 1995; Associate Chair, 1997-2007; Assoc. Prof., 1992-95; Asst. Prof. 1986-92.

Massachusetts Institute of Technology. Visiting Professor, July-December 2013.

INRIA (Institut National de Recherche en Informatique et en Automatique), Rocquencourt, France. Visiting Professor, academic year 2005-06 & summers 2004, 2007.

Bell Laboratories, Murray Hill, NJ. Member of Technical Staff, Summer 1984. Consultant, 1983-2001.

Carnegie-Mellon University, Pittsburgh, PA. Research and teaching assistant, 1982-85.

College of Medicine, University of Illinois, Urbana, IL. Computer programmer, summers 1976-80.

Awards and Honors

Kusaka Memorial Prize in Physics, Princeton University, 1981.

National Science Foundation Graduate Student Fellowship, 1981-1984.

ACM Fellow (Association for Computing Machinery), 1998.

The Other Prize, Programming Contest of the ACM International Conference on Functional Programming, 1998.

ACM SIGPLAN Distinguished Service Award, 2002.

ACM SIGPLAN selected "Real-time Concurrent Collection on Stock Multiprocessors" (Appel, Ellis, Li 1988) as one of the 50 most influential papers in 20 years of the PLDI conference, 2002.

Professional Activities

1. Program Committee, *ACM SIGPLAN '89 Conf. on Prog. Lang. Design and Implementation*, 1989.
2. Program Committee, *Seventeenth ACM Symp. on Principles of Programming Languages*, 1990.
3. Associate Editor, *ACM Transactions on Programming Languages and Systems*, 1990-1992.
4. Associate Editor, *ACM Letters on Programming Languages and Systems*, 1991-1992.
5. Program Chair, *Nineteenth ACM Symp. on Principles of Programming Languages*, 1992.
6. Co-editor, *Journal of Functional Programming* special issue on ML, 1992.
7. Program Committee, *Sixth ACM Conf. on Functional Prog. Lang. and Computer Architecture*, 1993.
8. Editor in Chief, *ACM Transactions on Programming Languages and Systems*, 1993-97.
9. Program Committee, *International Conference on Functional Programming*, 1997.
10. General Chair, *POPL'99: 26th ACM Symp. on Principles of Programming Languages*, 1999.
11. Program Committee, *IEEE Symposium on Security and Privacy*, 2002.
12. Program Committee, *ACM SIGPLAN Workshop on Types in Language Design and Implementation*, 2003.
13. Program Committee, *Nineteenth Annual IEEE Symposium on Logic in Computer Science*, 2004.
14. Program Committee, *ACM SIGPLAN 2005 Conference on Programming Language Design and Implementation (PLDI)*, 2005.
15. Program Committee, *International Workshop on Logical Frameworks and Meta-Languages: Theory and Practice (LFMTP'06)*, 2006.
16. Program Committee, *EVT'07: 2007 Usenix/ACCURATE Electronic Voting Technology Workshop*.
17. Program Committee, *POPL'09: 36th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, 2009.
18. Program Committee, *PLDI 2011: 32nd ACM SIGPLAN conference on Programming Language Design and Implementation*, 2011.
19. General Co-Chair, *ITP 2012: Interactive Theorem Proving*, 2012.
20. Program Committee, *POPL 2014: 41st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, 2014.
21. Award Committee, *SIGPLAN Programming Languages Software Award*, 2016.
22. Board of Advisors, *Verified Voting Foundation*, since 2015.
23. Program Committee, *POPL 2020: 47th ACM SIGPLAN Symposium on Principles of Programming Languages*, 2020.

Research Grants

1. *Implementation of an efficient reducer for lambda expressions*, National Science Foundation DCR-8603453, \$115,799, 1986-88.
2. Digital Equipment Corporation Faculty Incentive Grant, \$180,000, 1986-89.

3. *Unifying compile-time and run-time evaluation*, National Science Foundation CCR-8806121, \$123,510, 1988-90.
4. *Standard ML of New Jersey software capitalization*, National Science Foundation CCR-8914570, \$119,545, 1990-91.
5. *Using immutable types for debugging and parallelism*, National Science Foundation CCR-9002786, \$174,618, 1990-92.
6. *Optimization of space usage*, National Science Foundation CCR-9200790, \$348,119, 1992-96.
7. *Framework, Algorithms, and Applications for Cross-module Inlining*, National Science Foundation CCR-9625413, \$180,331, 1996-98.
8. *Development of a HIL/LIL Framework for a National Compiler Infrastructure*, Defense Advanced Research Projects Agency and National Science Foundation (as subcontractor to Univ. of Virginia), \$1,397,293, 1996-99.
9. *Tools, Interfaces, and Access Control for Secure Programming*, National Science Foundation CCR-9870316, \$322,000, 1998-2001 (co-PI).
10. *Scaling Proof-Carrying Code to Production Compilers and Security Policies*, Defense Advanced Research Projects Agency, \$3,870,378, 1999-2004.
11. *Applying Compiler Techniques to Proof-Carrying Code*, National Science Foundation CCR-9974553, \$220,000, 1999-2002.
12. IBM University Partnership Program, \$40,000, 1999-2000.
13. *High-Assurance Common Language Runtime*, National Science Foundation CCR-0208601, \$400,000, 2002-2005.
14. *Assurance-Carrying Components*, Advanced Research and Development Agency contract NBCHC030106, \$759,910, 2003-05.
15. Sun Microsystems research grant, \$20,000, 2004.
16. *End-to-end source-to-object verification of interface safety*, National Science Foundation grant CCF-0540914, \$325,000, 2006-09.
17. *MulVAL Technologies Plan*, New Jersey Commission on Science and Technology, \$60,000, 2006.
18. Microsoft Corporation research grant, \$25,000, 2006.
19. *Evidence-based Trust in Large-scale MLS Systems*, Air Force Office of Scientific Research FA9550-09-1-0138 (as subcontractor to Kansas State University), \$1,000,000, 2009-14.
20. *Combining Foundational and Lightweight Formal Methods to Build Certifiably Dependable Software*, National Science Foundation grant CNS-0910448, \$500,000, 2009-13.
21. *CARS: A Platform for Scaling Formal Verification to Component-Based Vehicular Software Stacks*, Defense Advanced Research Projects Agency award FA8750-12-2-0293, \$6,108,346, 2012-2017.
22. *Verified HMAC*, Google Advanced Technology and Projects grant, \$95,928, 2014.
23. *Principled Optimizing Compilation of Dependently Typed Languages*, National Science Foundation grant CCF-1407794, \$600,000, 2014-17.
24. *Concurrent separation logic for C*, Intel Corporation research grant, \$238,015, 2015-16.
25. *Collaborative Research: Expeditions in Computing: The Science of Deep Specification*, National Science Foundation grant CCF-1521602, \$3,453,419, 2015-20.

Publications

Books, chapters in books



1. "Garbage Collection," in *Topics in Advanced Language Implementation*, Peter Lee, ed. MIT Press, 1991.
2. *Compiling with Continuations*, Cambridge University Press, 1992.
3. *Modern Compiler Implementation in ML*, Cambridge University Press, 1998.
4. *Modern Compiler Implementation in Java*, Cambridge University Press, 1998.
5. *Modern Compiler Implementation in C*, Cambridge University Press, 1998.
6. *Modern Compiler Implementation in Java, 2nd edition*, with Jens Palsberg, Cambridge University Press, 2002.
7. *Alan Turing's Systems of Logic: The Princeton Thesis*, edited and introduced by Andrew W. Appel, Princeton University Press, 2012.
8. *Program Logics for Certified Compilers*, by Andrew W. Appel with Robert Dockins, Aquinas Hobor, Lennart Beringer, Josiah Dodds, Gordon Stewart, Sandrine Blazy, and Xavier Leroy. Cambridge University Press, 2014.
9. *Verified Functional Algorithms*, by Andrew W. Appel, 2017. Volume 3 of *Software Foundations*, edited by B. C. Pierce.

Journal papers, refereed conference papers, and patents

10. A Microprocessor-Based CAI System with Graphic Capabilities, by Frank J. Mabry, Allan H. Levy, and Andrew W. Appel, *Proc. 1978 conference, Assoc. for Development of Computer-based Instruction Systems*.
11. Rogomatic: A Belligerent Expert System, by Michael L. Mauldin, Guy J. Jacobson, Andrew W. Appel, and Leonard G. C. Hamer. *Proc. Fifth Nat. Conf. Canadian Soc. for Computational Studies of Intelligence*, May 1984.
12. An Efficient Program for Many-Body Simulations. *SIAM Journal on Scientific and Statistical Computing* 6(1):85-103, 1985.
13. Semantics-Directed Code Generation, by Andrew W. Appel, *Proc. Twelfth ACM Symposium on Principles of Programming Languages*, January 1985.
14. Generalizations of the Sethi-Ullman algorithm for register allocation. Andrew W. Appel and Kenneth J. Supowit, *Software Practice and Experience* 17(6):417-421, 1987.

15. A Standard ML compiler, by Andrew W. Appel and David B. MacQueen, *Proc. Third Int'l Conf. on Functional Programming & Computer Architecture (LNCS 274, Springer-Verlag)*, Portland, Oregon, September 1987.
16. Garbage collection can be faster than stack allocation. Andrew W. Appel. *Information Processing Letters* 25(4):275-279, 17 June 1987.
17. Real-time concurrent collection on stock multiprocessors, by Andrew W. Appel, John Ellis, and Kai Li, *Proc. ACM SIGPLAN '88 Conf. on Prog. Lang. Design & Implementation*, pp. 11-20, June 1988.
18. The World's Fastest Scrabble Program. Andrew W. Appel and Guy J. Jacobson, *Comm. ACM* 31(5):572-578, 585, May 1988.
19. Simulating digital circuits with one bit per wire. Andrew W. Appel, *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems* 7(9):987-993, September 1988.
20. Continuation-passing, closure-passing style, by Andrew W. Appel and Trevor Jim, *Proc. Sixteenth ACM Symposium on Principles of Programming Languages*, pp. 293-302, January 1989.
21. Simple Generational Garbage Collection and Fast Allocation. Andrew W. Appel. *Software--Practice and Experience* 19(2):171-183, February 1989.
22. Allocation without Locking. Andrew W. Appel. *Software--Practice and Experience* 19(7):703-705, July 1989.
23. Runtime Tags Aren't Necessary. Andrew W. Appel. *Lisp and Symbolic Computation* 2, 153-162 (1989).
24. Vectorized Garbage Collection. Andrew W. Appel and Aage Bendiksen. *The Journal of Supercomputing* 3, 151-160 (1989).
25. A Runtime System. *Lisp and Symbolic Computation* 3, 343-380, 1990.
26. An advisor for flexible working sets, by Rafael Alonso and Andrew W. Appel, *1990 ACM SIGMETRICS Conf. on Measurement and Modeling of Computer Systems*, pp. 153-162, May 1990.
27. Debugging Standard ML without reverse engineering, by Andrew P. Tolmach and Andrew W. Appel, *Proc. 1990 ACM Conf. on Lisp and Functional Programming*, pp. 1-12, June 1990.
28. Real-time concurrent garbage collection system and method, by John R. Ellis, Kai Li, and Andrew W. Appel. U.S. Patent 5,088,036, 1992.
29. Virtual memory primitives for user programs, by Andrew W. Appel and Kai Li, *Proc. Fourth Int'l Conf. on Architectural Support for Prog. Languages and Operating Systems*, (SIGPLAN Notices 26(4)) pp. 96-107, April 1991.
30. Standard ML of New Jersey, by Andrew W. Appel and David B. MacQueen, *Third Int'l Symp. on Prog. Lang. Implementation and Logic Programming, Springer-Verlag LNCS 528*, pp. 1-13, August 1991.
31. Callee-save registers in Continuation-Passing Style, by Andrew W. Appel and Zhong Shao. *Lisp and Symbolic Computation* 5, 189-219, 1992.
32. Smartest Recompilation, by Zhong Shao and Andrew W. Appel, *Proc. Twentieth ACM Symp. on Principles of Programming Languages*, January 1993.
33. A Critique of Standard ML. Andrew W. Appel. *Journal of Functional Programming* 3 (4) 391-430, 1993.
34. Unrolling Lists, by Zhong Shao, John H. Reppy, and Andrew W. Appel, *Proc. 1994 ACM Conf. on Lisp and Functional Programming*, pp. 185-195, June 1994.
35. Space-Efficient Closure Representations, by Zhong Shao and Andrew W. Appel, *Proc. 1994 ACM Conf. on Lisp and Functional Programming*, pp. 150-161, June 1994.
36. Separate Compilation for Standard ML, by Andrew W. Appel and David B. MacQueen, *Proc. 1994 ACM Conf. on Programming Language Design and Implementation (SIGPLAN Notices v. 29 #6)*, pp. 13-23, June 1994.

37. Axiomatic Bootstrapping: A guide for compiler hackers, Andrew W. Appel, *ACM Transactions on Programming Languages and Systems*, vol. 16, number 6, pp. 1699-1718, November 1994.
38. Loop Headers in Lambda-calculus or CPS. Andrew W. Appel. *Lisp and Symbolic Computation* 7, 337-343, 1994.
39. A Debugger for Standard ML. Andrew Tolmach and Andrew W. Appel. *Journal of Functional Programming*, vol. 5, number 2, pp. 155-200, April 1995.
40. A Type-Based Compiler for Standard ML, by Zhong Shao and Andrew W. Appel, *Proc. 1995 ACM Conf. on Programming Language Design and Implementation* (SIGPLAN Notices v. 30 #6), pp. 116-129, June 1995.
41. Cache Performance of Fast-Allocating Programs, by Marcelo J. R. Goncalves and Andrew W. Appel, *Proc. Seventh Int'l Conf. on Functional Programming and Computer Architecture*, pp. 293-305, ACM Press, June 1995.
42. Empirical and Analytic Study of Stack versus Heap Cost for Languages with Closures. Andrew W. Appel and Zhong Shao. *Journal of Functional Programming* 6 (1) 47-74, 1996.
43. How to Edit a Journal by E-mail. Andrew W. Appel *Journal of Scholarly Publishing* 27 (2) 82-99, January 1996.
44. Iterated Register Coalescing, by Lal George and Andrew W. Appel, *23rd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages* pp. 208-218, January 1996.
45. Iterated Register Coalescing. Lal George and Andrew W. Appel. *ACM Transactions on Programming Languages and Systems* 18(3) 300-324, May 1996. Shorter version appeared in *23rd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, January 1996.
46. Security and document compatibility for electronic refereeing. Andrew W. Appel. *CBE Views* 20(1), 1997, published by the Council of Biology Editors.
47. Lambda-Splitting: A Higher-Order Approach to Cross-Module Optimizations, by Matthias Blume and Andrew W. Appel, *Proc. ACM SIGPLAN International Conference on Functional Programming (ICFP '97)*, pp. 112-124, June 1997.
48. The Zephyr Abstract Syntax Description Language, by Daniel C. Wang, Andrew W. Appel, Jeff L. Korn, and Christopher S. Serra. *Conference on Domain-Specific Languages*, USENIX Association, October 1997.
49. Shrinking Lambda Expressions in Linear Time. Andrew W. Appel and Trevor Jim. *Journal of Functional Programming* v. 7 no. 5, pp. 515-540, 1997.
50. Traversal-based Visualization of Data Structures, by Jeffrey L. Korn and Andrew W. Appel, *IEEE Symposium on Information Visualization (InfoVis '98)*, pp. 11-18, October 1998.
51. Hierarchical Modularity. Matthias Blume and Andrew W. Appel, *ACM Transactions on Programming Languages and Systems*, 21 (4) 812-846, July 1999.
52. Lightweight Lemmas in Lambda Prolog, by Andrew W. Appel and Amy Felty, *16th International Conference on Logic Programming*, pp. 411-425, MIT Press, November 1999.
53. Proof-Carrying Authentication, by Andrew W. Appel and Edward Felten, *6th ACM Conference on Computer and Communications Security*, November 1999.
54. Efficient and Safe-for-Space Closure Conversion, Zhong Shao and Andrew W. Appel, *ACM Trans. on Prog. Lang. and Systems* 22(1) 129-161, January 2000.
55. A Semantic Model of Types and Machine Instructions for Proof-Carrying Code, by Andrew W. Appel and Amy P. Felty. *27th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL '00)*, pp. 243-253, January 2000.
56. Machine Instruction Syntax and Semantics in Higher Order Logic, by Neophytos G. Michael and Andrew W. Appel, *17th International Conference on Automated Deduction (CADE-17)*, Springer-Verlag (Lecture Notes in Artificial Intelligence), pp. 7-24, June 2000.
57. Technological Access Control Interferes with Noninfringing Scholarship. Andrew W. Appel and Edward W. Felten. *Communications of the ACM* 43 (9) 21-23, September 2000.

58. An Indexed Model of Recursive Types for Foundational Proof-Carrying Code. Andrew W. Appel and David McAllester. *ACM Transactions on Programming Languages and Systems* 23 (5) 657-683, September 2001.
59. Type-Preserving Garbage Collectors, Daniel C. Wang and Andrew W. Appel, *POPL 2001: The 28th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pp. 166-178, January 2001.
60. SAFKASI: A Security Mechanism for Language-Based Systems, Dan S. Wallach, Andrew W. Appel, and Edward W. Felten. *ACM Transactions on Software Engineering and Methodology*, 9 (4) 341-378, October 2000.
61. Optimal Spilling for CISC Machines with Few Registers, by Andrew W. Appel and Lal George. *ACM SIGPLAN 2001 Conference on Programming Language Design and Implementation*, pp. 243-253, June 2001.
62. Foundational Proof-Carrying Code, by Andrew W. Appel, *16th Annual IEEE Symposium on Logic in Computer Science (LICS '01)*, pp. 247-258, June 2001.
63. A Stratified Semantics of General References Embeddable in Higher-Order Logic, by Amal Ahmed, Andrew W. Appel, and Roberto Virga. *17th Annual IEEE Symposium on Logic in Computer Science (LICS 2002)*, pp. 75-86, June 2002.
64. Creating and Preserving Locality of Java Applications at Allocation and Garbage Collection Times, by Yefim Shuf, Manish Gupta, Hubertus Franke, Andrew W. Appel, and Jaswinder Pal Singh. *17th Annual ACM Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA 2002)*, *SIGPLAN Notices* 37(11) pp. 13-25, November 2002.
65. Mechanisms for secure modular programming in Java, by Lujo Bauer, Andrew W. Appel, and Edward W. Felten. *Software--Practice and Experience* 33:461-480, 2003.
66. A Trustworthy Proof Checker, by Andrew W. Appel, Neophytos G. Michael, Aaron Stump, and Roberto Virga. *Journal of Automated Reasoning* 31:231-260, 2003.
67. Using Memory Errors to Attack a Virtual Machine, by Sudhakar Govindavajhala and Andrew W. Appel, *2003 IEEE Symposium on Security and Privacy*, pp. 154-165, May 2003.
68. A Provably Sound TAL for Back-end Optimization, by Juan Chen, Dinghao Wu, Andrew W. Appel, and Hai Fang. *PLDI 2003: ACM SIGPLAN Conference on Programming Language Design and Implementation*, pp. 208-219, June 2003.
69. Foundational Proof Checkers with Small Witnesses, by Dinghao Wu, Andrew W. Appel, and Aaron Stump. *5th ACM-SIGPLAN International Conference on Principles and Practice of Declarative Programming*, pp. 264-274, August 2003.
70. Policy-Enforced Linking of Untrusted Components (Extended Abstract), by Eunyoung Lee and Andrew W. Appel, *European Software Engineering Conference and ACM SIGSOFT Symposium on the Foundations of Software Engineering*, pp. 371-374, September 2003.
71. Polymorphic Lemmas and Definitions in Lambda Prolog and Twelf, by Andrew W. Appel and Amy P. Felty. *Theory and Practice of Logic Programming* 4 (1) 1-39, January 2004.
72. Dependent Types Ensure Partial Correctness of Theorem Provers, by Andrew W. Appel and Amy P. Felty. *Journal of Functional Programming* 14(1):3-19, January 2004.
73. Construction of a Semantic Model for a Typed Assembly Language, by Gang Tan, Andrew W. Appel, Kedar N. Swadi, and Dinghao Wu. In *5th International Conference on Verification, Model Checking, and Abstract Interpretation (VMCAI '04)*, January 2004.
74. MulVAL: A Logic-based Network Security Analyzer by Xinming Ou, Sudhakar Govindavajhala, and Andrew W. Appel, In *14th Usenix Security Symposium*, August 2005.
75. A Compositional Logic for Control Flow by Gang Tan and Andrew W. Appel, in *7th International Conference on Verification, Model Checking, and Abstract Interpretation (VMCAI)*, January 2006.
76. Safe Java Native Interface, by Gang Tan, Andrew W. Appel, Srimat Chakradhar, Anand Raghunathan, Srivaths Ravi, and Daniel Wang. *International Symposium on Secure Software Engineering*, March 2006.

77. A Very Modal Model of a Modern, Major, General Type System, by Andrew W. Appel, Paul-Andre Mellies, Christopher D. Richards, and Jerome Vouillon. *POPL 2007: The 34th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, January 2007.
78. Separation Logic for Small-step C minor, by Andrew W. Appel and Sandrine Blazy, in *TPHOLs 2007: 20th International Conference on Theorem Proving in Higher-Order Logics*, pp. 5-21, September 2007.
79. Oracle Semantics for Concurrent Separation Logic, by Aquinas Hobor, Andrew W. Appel, and Francesco Zappa Nardelli, in *ESOP'08: European Symposium on Programming*, April 2008.
80. Multimodal Separation Logic for Reasoning About Operational Semantics, by Robert Dockins, Andrew W. Appel, and Aquinas Hobor, in *Twenty-fourth Conference on the Mathematical Foundations of Programming Semantics*, May 2008.
81. The New Jersey Voting-machine Lawsuit and the AVC Advantage DRE Voting Machine, by Andrew W. Appel, Maia Ginsburg, Harri Hursti, Brian W. Kernighan, Christopher D. Richards, Gang Tan, and Penny Venetis. In *EVT/WOTE'09, 2009 Electronic Voting Technology Workshop / Workshop on Trustworthy Elections*, August 2009.
82. A Fresh Look at Separation Algebras and Share Accounting by Robert Dockins, Aquinas Hobor, and Andrew W. Appel. *Seventh Asian Symposium on Programming Languages and Systems (APLAS 2009)*, December 2009.
83. A Theory of Indirection via Approximation, by Aquinas Hobor, Robert Dockins, and Andrew W. Appel. *POPL 2010: The 37th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pp. 171-184, January 2010.
84. Formal Verification of Coalescing Graph-Coloring Register Allocation, by Sandrine Blazy, Benoit Robillard and Andrew W. Appel. *ESOP 2010: 19th European Symposium on Programming*, pp. 145-164, March 2010.
85. Concurrent Separation Logic for Pipelined Parallelization, by Christian J. Bell, Andrew W. Appel, and David Walker. In *SAS 2010: 17th Annual Static Analysis Symposium*, September 2010.
86. Semantic Foundations for Typed Assembly Languages, by A. Ahmed, A. W. Appel, C. D. Richards, K. Swadi, G. Tan, and D. C. Wang. *ACM Transactions on Programming Languages and Systems*, 32(3):7.1-7.67, March 2010.
87. A Logical Mix of Approximation and Separation by Aquinas Hobor, Robert Dockins, and Andrew W. Appel. In *APLAS 2010: 8th ASIAN Symposium on Programming Languages and Systems*, November 2010.
88. Local Actions for a Curry-style Operational Semantics by Gordon Stewart and Andrew W. Appel. In *PLPV'11: 5th ACM SIGPLAN Workshop on Programming Languages meets Program Verification*, January 29, 2011.
89. Verified Software Toolchain, by Andrew W. Appel. In *ESOP 2011: 20th European Symposium on Programming*, LNCS 6602, pp. 1-17, March 2011.
90. VeriSmall: Verified Smallfoot Shape Analysis, by Andrew W. Appel. In *CPP 2011: First International Conference on Certified Programs and Proofs*, Springer LNCS 7086, pp. 231-246, December 2011.
91. A Certificate Infrastructure for Machine-Checked Proofs of Conditional Information Flow, by Torben Amtoft, Josiah Dodds, Zhi Zhang, Andrew Appel, Lennart Beringer, John Hatcliff, Xinming Ou and Andrew Cousino. *First Conference on Principles of Security and Trust (POST 2012)*, LNCS 7215, pp. 369-389, March 2012.
92. A list-machine benchmark for mechanized metatheory by Andrew W. Appel, Robert Dockins, and Xavier Leroy. *Journal of Automated Reasoning* 49(3):453-491, 2012. DOI 10.1007/s10817-011-9226-1
93. Security Seals On Voting Machines: A Case Study, by Andrew W. Appel. *ACM Transactions on Information and System Security (TISSEC)* 14 (2) pages 18:1--18:29, September 2011.

94. Verified Heap Theorem Prover by Paramodulation, by Gordon Stewart, Lennart Beringer, and Andrew W. Appel. In *ICFP 2012: The 17th ACM SIGPLAN International Conference on Functional Programming*, pp. 3-14, September 2012.
95. Mostly Sound Type System Improves a Foundational Program Verifier, by Josiah Dodds and Andrew W. Appel. *3rd International Conference on Certified Programs and Proofs (CPP 2013)*, December 2013.
96. Verified Compilation for Shared-memory C, by Lennart Beringer, Gordon Stewart, Robert Dockins, and Andrew W. Appel. *ESOP'14: 23rd European Symposium on Programming*, April 2014.
97. Portable Software Fault Isolation, by Joshua A. Kroll, Gordon Stewart, and Andrew W. Appel. *CSF'14: Computer Security Foundations Symposium*, IEEE Press, July 2014.
98. Compositional CompCert, by Gordon Stewart, Lennart Beringer, Santiago Cuellar, and Andrew W. Appel. *POPL 2015: The 42nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 275-287, January 2015.
99. Verified Correctness and Security of OpenSSL HMAC, by Lennart Beringer, Adam Petcher, Katherine Q. Ye, and Andrew W. Appel. In *24th USENIX Security Symposium*, pages 207-221, August 2015.
100. Verification of a Cryptographic Primitive: SHA-256, by Andrew W. Appel. *ACM Transactions on Programming Languages and Systems*, 37(2) 7:1-7:31, April 2015.
101. Modular Verification for Computer Security, by Andrew W. Appel, in *29th IEEE Computer Security Foundations Symposium (CSF'16)*, June 2016.
102. Shrink Fast Correctly! by Olivier Savary Belanger and Andrew W. Appel. *Proceedings of International Symposium on Principles and Practice of Declarative Programming (PPDP'17)*, 12 pages, October 2017 (PPDP'17).
103. Verified Correctness and Security of mbedTLS HMAC-DRBG by Katherine Q. Ye, Matthew Green, Naphat Sanguansin, Lennart Beringer, Adam Petcher, and Andrew W. Appel. *CCS'17: ACM Conference on Computer and Communications Security*, October 2017.
104. Bringing order to the separation logic jungle, by Qinxiang Cao, Santiago Cuellar, and Andrew W. Appel. *APLAS'17: 15th Asian Symposium on Programming Languages and Systems*, November 2017.
105. A verified messaging system, by William Mansky, Andrew W. Appel, and Aleksey Nogin. *OOPSLA'17: ACM Conference on Object-Oriented Programming Systems, Languages, and Applications*, October 2017. *Proceedings of the ACM on Programming Languages (PACM/PL)* volume 1, issue OOPSLA, paper 87, 2017.
106. Position paper: the science of deep specification, by Andrew W. Appel, Lennart Beringer, Adam Chlipala, Benjamin C. Pierce, Zhong Shao, Stephanie Weirich and Steve Zdancewic, *Philosophical Transactions of the Royal Society A* 375:21060331 (24 pages), 2017.
107. VST-Floyd: A separation logic tool to verify correctness of C programs, by Qinxiang Cao, Lennart Beringer, Samuel Gruetter, Josiah Dodds, and Andrew W. Appel. *Journal of Automated Reasoning* 61(1), pp. 367-422, 2018. (Local copy)
108. Closure Conversion is Safe for Space, by Zoe Paraskevopoulou and Andrew W. Appel. *Proceedings of the ACM on Programming Languages*, vol. 3, no. ICFP, article 83, 29 pages, doi 10.1145/3341687, August 2019.
109. Abstraction and Subsumption in Modular Verification of C Programs, by Lennart Beringer and Andrew W. Appel. *FM2019: 23rd International Symposium on Formal Methods*, October 2019.
110. Connecting Higher-Order Separation Logic to a First-Order Outside World, by William Mansky, Wolf Honoré, and Andrew W. Appel, *ESOP 2020: European Symposium on Programming*, April 2020.
111. Ballot-Marking Devices (BMDs) Cannot Assure the Will of the Voters, by Andrew W. Appel, Richard A. DeMillo, and Philip B. Stark, February 2020. Accepted for publication in *Election Law Journal*. (Earlier versions appeared on SSRN.)

112. Verified sequential malloc/free, by Andrew W. Appel and David A. Naumann, (to appear) in *2020 ACM SIGPLAN International Symposium on Memory Management*, June 2020.

Workshop and unrefereed conference papers

113. Debuggable concurrency extensions for Standard ML, by Andrew P. Tolmach and Andrew W. Appel, *Proc. ACM/ONR Workshop on Parallel and Distributed Debugging*, May 1991 (SIGPLAN Notices, Dec. 1991), pp. 115-127.
114. Efficient Substitution in Hoare Logic Expressions, by Andrew W. Appel, Kedar Swadi, and Roberto Virga. *4th International Workshop on Higher-Order Operational Techniques in Semantics (HOOTS 2000)*, pp. 35-50, September 2000.
115. Fair use, public domain, or piracy ... should the digital exchange of copyrighted works be permitted or prevented? (Rountable Panel II: Digital Video), by Andrew W. Appel, Jeffrey Cunard, Martin Garbus, and Edward Hernstadt, *Fordham Intellectual Property, Media & Entertainment Law Journal*, volume 11, number 2, page 317, 2001.
116. A Trustworthy Proof Checker, by Andrew W. Appel, Neophytos G. Michael, Aaron Stump, and Roberto Virga. In *Verification Workshop - VERIFY 2002* and (jointly) in *Foundations of Computer Security - FCS 2002* Copenhagen, Denmark, July 25-26, 2002.
117. A list-machine benchmark for mechanized metatheory (extended abstract) by Andrew W. Appel and Xavier Leroy. *LFMTP'06: International Workshop on Logical Frameworks and Meta-Languages: Theory and Practice*, August 2006.
118. Effective Audit Policy for Voter-Verified Paper Ballots, presented at *2007 Annual Meeting of the American Political Science Association*, Chicago, September 1, 2007.

Review Articles, Tutorials, Position Papers

119. Book Review of *Garbage Collection: Algorithms for Automatic Dynamic Memory Management* by Richard Jones and Rafael Lins. *Journal of Functional Programming* 7(2), pp. 227-229, March 1997.
120. SSA is Functional Programming. *ACM SIGPLAN Notices* v. 33, no. 4, pp. 17-20, April 1998.
121. Protection against untrusted code. *IBM Developer Works*, September 1999.
122. Retrospective: Real-time Concurrent Collection on Stock Multiprocessors. *20 Years of the ACM/SIGPLAN Conference on Programming Language Design and Implementation (1979-1999): A Selection*, ACM Press, 2004.
123. Foundational High-level Static Analysis. In *CAV 2008 Workshop on Exploiting Concurrency Efficiently and Correctly*, July 2008.
124. Technical Perspective: The Scalability of CertiKOS, by Andrew W. Appel, *Communications of the ACM*, vol. 62 no.10, page 88. DOI 10.1145/335690610.1145/3356906.
125. *Freedom-to-Tinker*: 16 articles on the freedom-to-tinker.com blog between 2007 and 2009; 6 articles in 2010; 15 articles in 2011.
126. The Birth of Computer Science at Princeton in the 1930s, in A. W. Appel, ed., *Alan Turing's Systems of Logic: The Princeton Thesis*, Princeton University Press, 2012.
127. Research Needs for Secure, Trustworthy, and Reliable Semiconductors, by Andrew Appel, Chris Daverse, Kenneth Hines, Rafic Makki, Keith Marzullo, Celia Merzbacher, Ron Perez, Fred Schneider, Mani Soma, and Yervant Zorian. Final workshop report of the NSF/CCC/SRC workshop on Convergence of Software Assurance Methodologies and Trustworthy Semiconductor Design and Manufacture, 2013.
128. CertiCoq: A verified compiler for Coq, by Abhishek Anand, Andrew Appel, Greg Morrisett, Zoe Paraskevopoulou, Randy Pollack, Olivier Savary Belanger, Matthieu Sozeau, and Matthew

- Weaver. In *CoqPL'17: The Third International Workshop on Coq for Programming Languages*, January 2017.
129. Position paper: the science of deep specification, by Andrew W. Appel, Lennart Beringer, Adam Chlipala, Benjamin C. Pierce, Zhong Shao, Stephanie Weirich, Steve Zdancewic. *Philosophical Transactions of the Royal Society A* vol. 375, no. 2104, September 2017.
 130. *Securing the Vote: Protecting American Democracy*, by National Academies of Science, Engineering, and Medicine: Lee C. Bollinger, Michael A. McRobbie, Andrew W. Appel, Josh Benaloh, Karen Cook, Dana DeBeauvoir, Moon Duchin, Juan E. Gilbert, Susan L. Graham, Neal Kelley, Kevin J. Kennedy, Nathaniel Persily, Ronald L. Rivest, Charles Stewart III. September 2018.
 131. Evidence-Based Elections: Create a Meaningful Paper Trail, then Audit, by Andrew W. Appel and Philip B. Stark, (to appear) in *Georgetown Law Technology Review*, 2020.

Unrefereed papers

132. An Investigation of Galaxy Clustering Using an Asymptotically Fast N-Body Algorithm. Senior Thesis, Princeton University, 1981.
133. Compile-time Evaluation and Code Generation in Semantics-Directed Compilers. Ph.D. Thesis, Carnegie-Mellon University, July 1985.
134. Concise specifications of locally optimal code generators, Princeton Univ. Dept. of Computer Science CS-TR-080-87, 1987.
135. Re-opening closures, Princeton Univ. Dept. of Computer Science CS-TR-079-87, February 1987.
136. Optimizing closure environment representations, by Andrew W. Appel and Trevor Jim. Princeton Univ. Dept. of Computer Science CS-TR-168-88, July 1988.
137. Unifying Exceptions with Constructors in Standard ML, with David MacQueen, Robin Milner, and Mads Tofte. Univ. of Edinburgh Dept. of Comp. Sci. CSR-266-88, May 1988.
138. Profiling in the presence of optimization and garbage collection, by Andrew W. Appel, Bruce Duba, and David MacQueen. CS-TR-197-88, November 1988.
139. Hash-Consing Garbage Collection, by Andrew W. Appel and Marcelo J.R. Goncalves, Technical report TR-412-93, Department of Computer Science, Princeton University, January 1993.
140. Emulating Write-Allocate on a No-Write-Allocate Cache, by Andrew W. Appel, CS-TR-459-94, Princeton University, June 20, 1994.
141. Is POPL Mathematics or Science?, by Andrew W. Appel, *ACM SIGPLAN Notices* 27 (4), pp. 87-89, April 1992.
142. Intensional Equality \Rightarrow for Continuations, by Andrew W. Appel, *ACM SIGPLAN Notices* 31 (2), pp. 55-57, February 1996.
143. Ceci n'est pas une urne: On the Internet vote for the *Assemblée des Français de l'Etranger*, by Andrew W. Appel, June 2006.
144. Insecurities and Inaccuracies of the Sequoia AVC Advantage 9.00H DRE Voting Machine, by Andrew W. Appel, Maia Ginsburg, Harri Hursti, Brian W. Kernighan, Christopher D. Richards, and Gang Tan. October 2008.
145. The CompCert Memory Model, Version 2, by Xavier Leroy, Andrew W. Appel, Sandrine Blazy, and Gordon Stewart. INRIA Research Report RR-7987, June 2012.
146. Compiler Correctness for Concurrency: from concurrent separation logic to shared-memory assembly language, by Santiago Cuellar, Nick Giannarakis, Jean-Marie Madiot, William Mansky, Lennart Beringer, and Andrew W. Appel, Technical report TR-014-19, Department of Computer Science, Princeton University, March 2020.

147. Fair Elections During a Crisis: Urgent Recommendations in Law, Media, Politics, and Tech to Advance the Legitimacy of, and the Public Confidence in, the November 2020 U.S. Elections., by the Ad Hoc Committee for 2020 Election Fairness and Legitimacy (Appel, Azari, Cain, *et al.*), edited by Richard L. Hasen, UCI Law School, April 2020.

PhD Students

1. Andrew P. Tolmach, Ph.D. (1992) *Debugging Standard ML*. Professor, Portland State University.
2. Zhong Shao, Ph.D. (1994) *Compiling Standard ML for Efficient Execution on Modern Machines*. Professor, Yale University.
3. Marcelo J. R. Goncalves, Ph.D. (1995) *Cache Performance of Programs with Intensive Heap Allocation and Generational Garbage Collection*.
4. Matthias Blume, Ph.D. (1997) *Hierarchical Modularity and Intermodule Optimization*. Computer Scientist, Google, Inc.
5. Richard (Drew) Dean, Ph.D. (1999) *Formal Aspects of Mobile Code Security*. Senior Computer Scientist, SRI International.
6. Jeffrey L. Korn, Ph.D. (1999) *Abstraction and Visualization in Graphical Debuggers*. Software Engineer, Google, Inc.
7. Daniel C. Wang, Ph.D. (2002) *Managing Memory with Types*. Computer Scientist, Amazon.com.
8. Kedar N. Swadi, Ph.D. (2003) *Typed Machine Language*. CTO, AlgoAnalytics, Pune, India.
9. Lujo Bauer, Ph.D. (2003) *Access Control for the Web via Proof-Carrying Authorization*. Associate Professor, Carnegie Mellon University.
10. Eunyoung Lee, Ph.D. (2003) *Secure Linking: A Logical Framework for Policy-Enforced Component Composition*. Associate Professor, Dongduk Women's University, Seoul, Korea.
11. Juan Chen, Ph.D. (2004) *A Low-Level Typed Assembly Language with a Machine-checkable Soundness Proof*. Computer Scientist, Google, Inc.
12. Amal J. Ahmed, Ph.D. (2004) *Semantics of Types for Mutable State*. Associate Professor, Northeastern University.
13. Gang Tan, Ph.D. (2005) *A Compositional Logic for Control Flow and its Application to Foundational Proof-Carrying Code*. Associate Professor, Pennsylvania State University.
14. Dinghao Wu, Ph.D. (2005) *Interfacing Compilers, Proof Checkers, and Proofs for Foundational Proof-Carrying Code*. Associate Professor, Pennsylvania State University.
15. Xinming Ou, Ph.D. (2005) *A Logic Programming Approach to Network Security Analysis*. Professor, University of South Florida.
16. Sudhakar Govindavajhala, Ph.D. (2006) *A Formal Approach to Practical Network Security Management*. Computer and network security consultant.
17. Aquinas Hobor, Ph.D. (2008) *Oracle Semantics*. Assistant Professor, National University of Singapore and Yale/NUS college.
18. Christopher D. Richards, Ph.D. (2010) *The Approximation Modality in Models of Higher-Order Types*. Computer Scientist, Google, Inc.
19. Robert Dockins, Ph.D. (2012) *Operational Refinement for Compiler Correctness*. Researcher, Galois.com.
20. James Gordon Stewart, Ph.D. (2015) *Verified Separate Compilation for C*. Assistant Professor, Ohio University.
21. Josiah Dodds, Ph.D. (2015) *Computation Improves Interactive Symbolic Execution*. Researcher, Galois.com.
22. Qinxiang Cao, Ph.D. (2018) *Separation-Logic-based Program Verification in Coq*. Assistant Professor, Shanghai Jiao Tong University.

23. Olivier Savary Bélanger, Ph.D. (2019) *Verified Extraction for Coq*. Researcher, Galois.com.
24. Santiago Cuellar, PhD (2020) *Concurrent Permission Machine for modular proofs of optimizing compilers with shared memory concurrency*. Researcher, Galois.com.

The documents linked from this page are included to ensure timely dissemination of scholarly and technical work on a non-commercial basis. Copyright and all rights therein are maintained by the authors or by other copyright holders, notwithstanding that they have offered their works here electronically. It is understood that all persons copying this information will adhere to the terms and constraints invoked by each author's copyright. These works may not be reposted without the explicit permission of the copyright holder.

EXHIBIT B

**IN THE UNITED STATES DISTRICT COURT
FOR THE NORTHERN DISTRICT OF GEORGIA
ATLANTA DIVISION**

**DONNA CURLING, ET AL.,
Plaintiffs,**

v.

**BRAD RAFFENSPERGER, ET AL.,
Defendants.**

Civil Action No. 1:17-CV-2989-AT

**DECLARATION OF ANDREW W. APPEL
IN SUPPORT OF MOTION FOR PRELIMINARY INJUNCTION**

ANDREW W. APPEL, declares, under penalty of perjury, pursuant to 28 U.S.C. § 1746, that the following is true and correct:

1. My name is Andrew W. Appel.
2. My background, qualifications, and professional affiliations are set forth in my curriculum vitae, which is attached as Exhibit A. I have over 40 years' experience in computer science, and 15 years' experience studying voting machines and elections.
3. I am the Eugene Higgins Professor of Computer Science at Princeton University, where I have been on the faculty since 1986 and served as Department Chair from 2009-2015. I have also served as Director of Undergraduate Studies, Director of Graduate Studies, and Associate Chair in that department. I have

served as Editor in Chief of ACM Transactions on Programming Languages and Systems, the leading journal in my field. In 1998 I was elected a Fellow of the Association for Computing Machinery, the leading scientific and professional society in Computer Science.

4. I received an A.B. (1981) from Princeton University *summa cum laude* in Physics, and a PhD (1985) from Carnegie Mellon University in Computer Science.

5. I have taught undergraduate and graduate courses at Princeton University in programming, programming languages, software engineering, election machinery, software verification, and formal methods.

6. I have testified on election technology before the U.S. House of Representatives (subcommittee on information technology, 2016), the New Jersey legislature (several committees, on several occasions 2005-2018), the Superior Court of New Jersey (Mercer County, 2009; Cumberland County, 2011), the New York State Board of Elections (2019), the Freeholders of Mercer County (2017 and 2019) and Essex County (2019).

7. I have published over 100 scientific articles and books, including many papers on computer security and several papers on voting machines, election technology, and election audits.

8. I have served as a peer-review referee for the Usenix Electronic Voting Technology workshop.

9. I am not being compensated for my work related to this matter. I expect that my expenses, if any, will be reimbursed.

III. Dr. Shamos is incorrect regarding paper ballot risks.

10. I have read the Declaration of Michael Shamos, Ph.D., J.D., Doc. No. 472-1 filed July 10, 2019 in the above-captioned matter.

11. In general, Dr. Shamos's arguments against paper ballots are either irrelevant to current technology or to Curling Plaintiffs' proposed relief, or they apply equally to systems already in use by Georgia.

12. In paragraph 36, Dr. Shamos writes, "the paper ballot is the only record of the voter's choices." This is untrue, especially in precinct-count optical scan (PCOS) voting—which I understand is the method of voting proposed in the relief requested by Curling Plaintiffs. Generally in PCOS voting, the voter feeds his or her ballot directly into the PCOS machine, which electronically scans it and records the vote choices (and modern PCOS machines also scan a high-resolution image of the entire page). Yes, the PCOS is a computer that can be hacked, and therefore the paper ballot marked by the voter should be the *presumptive* record of the vote, but the memory image in the PCOS can provide important forensic evidence if it is suspected that the paper ballots have been tampered with. In this way, the electronic record inside the PCOS (and also in the results cartridge

removed from the PCOS at the close of the election day) serve the same role they do in the DREs that Dr. Shamos discusses in his paragraph 37.

13. In paragraph 37, Dr. Shamos allows that memory cards of DREs can be tampered with, but claims that “such a manipulation does not change the redundant records that are retained on each individual voting machines, and does not change the paper tabulations that are produced at the close of polls in each individual polling place and signed by election judges. ... Any discrepancy would be investigated” He continues in paragraph 38, “if there is a discrepancy between optical scan totals and hand-counting, the hand-counted totals are always used in the naïve belief that they are more reliable....”

14. It is not clear to me why Dr. Shamos believes that discrepancies with DREs would be investigated, but with PCOS machines the election officials or the courts would be too naïve to investigate. Dr. Shamos cites nothing to support this conclusion, nor does this conclusion follow from common sense.

15. In paragraph 43, Dr. Shamos shows a large figure showing approximately 18 steps in the chain of custody of paper ballots in Los Angeles County. He neglects to mention that almost exactly the same 18 steps are required for the chain of custody of electronic vote cartridges produced by DRE machines.

16. In paragraph 44, Dr. Shamos alleges that “ballot boxes are completely out of view of the public or poll watchers for a substantial period of time” and

volunteers a hypothetical scenario in which a “political operative bribes an insider to stop off while transporting ballot boxes to a tabulation center” and replaces the actual ballots with “pre-prepared ballots marked to favor his party’s candidates.” These allegations completely ignore that Georgia rules provide for public setup, testing, tabulation, and consolidation of votes, and no laws nor practical hurdles prevent the public or poll-watchers from tracking the ballot boxes from the precincts to be tabulated.¹

17. Dr. Shamos’s claim that “[i]n every election cycle in the United States, ballot boxes are found weeks after the election in place (such as lakes and rivers) making it clear that they were never counted” (§ 39) is also inaccurate and misleading.

a. The first citation he offers (§ 39 n.4) shows that, in fact, the optical scanners made a complete count of the ballots that were only later left behind. Moreover, while the chain of custody for a single box of ballots may have been broken, the ballots were not lost, thereby preserving the ability to check the paper records against the electronic optical scan records.

b. Another citation Dr. Shamos offers (§ 39 n.7) is irrelevant to in-precinct ballot boxes, which I understand is the relief advocated by Curling

¹ See O.C.G.A. §§ 21-2-379.11(f), 21-2-408; Ga. Comp. R. & Regs. 183-1-12-.02(5)(a)(8), (c)(4).

Plaintiffs here. Rather, this citation is about ballot drop-boxes (used in some states as an alternative to returning absentee ballots by mail), which are not used in Georgia nor are they part of the Curling Plaintiffs' proposed relief. The security measures, access, and timing surrounding those drop-boxes are different and not comparable to the procedures in place for in-precinct voting.²

c. Dr. Shamos's citation regarding Broward County (§ 39 n.9) demonstrates that a careful recanvas better accounts for all ballots—consistent with the goal of the relief sought by Curling Plaintiffs.

d. Dr. Shamos cites fraud with paper absentee ballots in North Carolina (§ 39 n.10, and again in § 41) – but Georgia already employs paper absentee ballots. Presumably Dr. Shamos is not suggesting that this evidence is indicative of what happens in Georgia, in which case it would be a critical, ongoing problem in Georgia unrelated to Curling Plaintiffs' proposed relief.

18. In his paragraphs 55 and 56, Dr. Shamos implies that because optical scanners may interpret ballots differently than humans would during a recount, therefore some sort of problem ensues. But then in the first sentence of paragraph

² See, e.g., [https://govt.westlaw.com/calregs/Document/I118620FA785243678FC16FA7D8FF09BD?viewType=FullText&originationContext=documenttoc&transitionType=CategoryPageItem&contextData=\(sc.Default\)](https://govt.westlaw.com/calregs/Document/I118620FA785243678FC16FA7D8FF09BD?viewType=FullText&originationContext=documenttoc&transitionType=CategoryPageItem&contextData=(sc.Default)).

57, he says that in these cases a manual recount would get the true result intended by the voter. Exactly! This very point undermines Dr. Shamos's overall conclusions, and supports Curling Plaintiffs' proposed relief. Audits or recounts of papers ballots can correct for fraudulent hacking of voting machines *and* can correct for accidental miscalibration or misconfiguration of voting machines.

19. In paragraph 56, Dr. Shamos discusses optical-scan voting machines, but refers to an obsolete technology that has not been manufactured in the U.S. for over a decade. He refers to "each optical scan sensor (and there is one for each column of the ballot)." Such voting machines were made in the 20th century, but 21st-century optical-scan voting machines take a high-resolution digital scan of the ballot page and then use algorithms to interpret the voter's marks. Perhaps Dr. Shamos has not studied the last two generations of optical-scan voting machines certified by the E.A.C.

IV. Dr. Shamos's conclusions regarding DREs are incorrect

20. Contrary to Dr. Shamos's conclusions, Dr. Halderman's description of how DREs can be easily hacked is consistent with the scientific consensus, as described in peer-reviewed academic publications and in other venues, and agrees with my own research and study of this issue. Dr. Shamos's claims (of the supposed difficulty in hacking) in his paragraphs 87-90 and 92-95 are incorrect, unsupported

by scientific research, contradicted by the published scientific research, and inconsistent with the scientific consensus.

21. It is well understood by computer scientists that computers (since 1950) are “stored program” machines, that is, the program that determines how they compute is stored in the memory of the computer itself. Replacing this program with a different program will instruct the computer in a different way. Replacing a legitimate vote-counting program in a DRE with a different program that fraudulently miscounts the votes will instruct the computer to fraudulently miscount the votes. Installation of a fraudulent program can be done in the factory before the DRE is shipped, it can be done by anyone with physical access to the machine, and it can be done in other ways.

22. A DRE can be “hacked,” that is, its computer program in memory can be replaced by a fraudulent program, by an attacker who never even comes within 100 miles of the machine. Before every election, election workers must install “ballot definition cartridges” into each voting machine. These cartridges are programmed in the election-management computers of a state or of a county, or by a private contractor. The *same* cartridges and the *same* physical insertion method is used to install new vote-counting programs into the DRE; this provision was designed (by

the manufacturer) to support the “firmware upgrade” process, whereby voting machines can be “upgraded” in the field³.

23. An attacker who can hack into the election-management system can “hijack” the (otherwise legitimate) ballot-definition-insertion process and turn it into an (illegitimate) firmware-upgrade process. This method is documented in the scientific literature^{4 5 6}, and has been demonstrated in the laboratory.

24. Election-management computers must be routinely (directly or indirectly) connected to the internet (or to the phone system, which nowadays is the same thing) for a variety of purposes, including the dissemination of election results.

25. It is well-known (and documented) as a matter of science, and to anyone who reads the newspaper, that computer systems connected (directly or indirectly) to the internet are often hacked, that is, infiltrated by malicious attackers. Computers have been hacked that are owned and managed by businesses (including large and small retailers, insurance companies, phone companies, and internet companies) and governments (including the U.S. government, state governments, and municipalities).

³ This sentence characterizes most but not all DRE voting machines, and characterizes the DREs used in Georgia.

⁴ Security Analysis of the Diebold AccuVote-TS Voting Machine, by Ariel J. Feldman, J. Alex Halderman, and Edward W. Felten, *Proc. 2007 USENIX/ACCURATE Electronic Voting Technology Workshop (EVT '07)*, August 2007

⁵ Security evaluation of ES&S voting machines and election management system., by Aviv, A., Černý, P., Clark, S., Cronin, E., Shah, G., Sherr, M., & Blaze, M. *Proceedings of the conference on Electronic voting technology*, p. 11, July 2008.

⁶ The New Jersey Voting-machine Lawsuit and the AVC Advantage DRE Voting Machine, by Andrew W. Appel, Maia Ginsburg, Harri Hursti, Brian W. Kernighan, Christopher D. Richards, Gang Tan, and Penny Venetis. *EVT/WOTE'09, Electronic Voting Technology Workshop / Workshop on Trustworthy Elections*, August 2009.

26. In cases where the hacker has wished to be stealthy, in some cases the hacks have survived for many years without detection.^{7 8}

27. Therefore Dr. Shamos is incorrect in asserting that it would be impractical to hack Georgia's DREs by a fully remote attack, and that any such attack would be readily detected.

28. In paragraph 89, Dr. Shamos says, "the machines' software is tested by independent testing authorities." Such testing is irrelevant. That testing was performed once, long ago, on a few instances of the DREs. It is not performed on the DREs in the field, and therefore could not possibly detect any fraudulent software installed in those DREs.

29. Dr. Shamos, in his paragraphs 97-101, promotes and defends "parallel testing." Parallel testing would be an extremely labor-intensive and impractical means of detecting DRE fraud, if it were ever done as thoroughly as would be necessary to be reliable. There is no evidence that parallel testing has ever been done, in any state, at a large enough scale to reliably assure the absence of DRE hacking.

⁷ Inside the West's failed fight against China's 'Cloud Hopper' hackers, By Jack Stubbs, Joseph Menn, and Christopher Bing, Reuters News Service, June 26, 2019.

⁸ Hackers are stealing years of call records from hacked cell networks, by Zack Whittaker, techcrunch.com, June 24, 2019.

30. Chris Harvey, Director of Elections of the State of Georgia, in a letter⁹ dated August 1, 2018 to County Commissioners, describes an extremely ad-hoc and lightweight regime of parallel testing used by Georgia in 2018. Based on this description, I can say that Georgia does not do effective parallel testing of DREs.

V. Dr. Shamos's claims go against the weight of scientific consensus

31. In 2017 I was appointed by the National Academies of Science, Engineering, and Medicine (NASEM) to serve on a Consensus Study Committee on the Future of Voting. I served on that study committee, which comprised five computer scientists, one mathematician, two political scientists, one law professor, three election officials (from Wisconsin, Texas, and California), and was chaired by two university presidents (one a computer scientist by background, the other a law professor).

32. This NASEM study committee met for five two-day meetings over 16 months, heard testimony from many experts and election administrators, and drafted a comprehensive report in June 2018. This report was sent by NASEM for thorough peer review by a panel of 14 expert reviewers, overseen by a computer-science professor and a law professor who were independent of the study group.

⁹ <http://www.accg.org/docs/policy/8-1-18%20Letter%20from%20Chris%20Harvey%20to%20County%20Commissions.pdf>

After it passed peer review, it was released by NASEM in September 2018, with the title *Securing the Vote: Protecting American Democracy*.

33. The NASEM consensus study report makes many specific recommendations, backed up by lengthy scientific justification. Two of our key recommendations are:

4.11. Elections should be conducted with human-readable paper ballots. These may be marked by hand or by machine (using a ballot-marking device); they may be counted by hand or by machine (using an optical scanner). Recounts and audits should be conducted by human inspection of the human-readable portion of the paper ballots. Voting machines that do not provide the capacity for independent auditing (e.g., machines that do not produce a voter-verifiable paper audit trail) should be removed from service as soon as possible.

4.12. Every effort should be made to use human-readable paper ballots in the 2018 federal election. All local, state, and federal elections should be conducted using human-readable paper ballots by the 2020 presidential election.

34. Our report represents the true scientific consensus not only of the committee itself, but also (to the best of our ability) of the broader scientific community; and this consensus was also tested by the external peer reviewers, who would not have let non-consensus recommendations pass unchallenged.

35. In general (and specifically on these recommendations 4.11 and 4.12) the committee did not have difficulty reaching consensus or identifying the broad scientific consensus; the science here is clear.

36. Those members of our committee who are computer scientists have substantial expertise in computer security with applications to elections, and at least three have studies issues specifically pertaining to paper ballots and to the security of voting machines such as DREs and optical scanners. Those members of our committee who were state-level or county-level election officials have many years of experience administering elections with voting machines and paper ballots.

37. I have been studying voting machines, as a substantial part of my scientific work, since 2004. I have taught two courses on “Election Machinery” at Princeton University. I have written papers on the security analysis of DRE voting machines, on security seals for voting machines, on election auditing, on internet voting, on ballot-marking devices. I have also written 58 short articles (between 2008 and 2019) about voting machines and elections, on Princeton University’s “Freedom to Tinker: research and expert commentary on digital technologies in public life.”

38. During the period 2004-19 I have spoken with, or corresponded with, or read the work of, well over 100 experts on the computer science of voting machines. In preparing this declaration, I also reviewed the scientific literature since 2007 on voting machines, with publications by dozens of scientists. On these bases, I understand that, with one exception, all computer-science experts on voting machines recognize that voting machines are not difficult to reprogram (to

“hack” if reprogrammed without authorization) and therefore it is unacceptably insecure to use paperless DRE voting machines in public elections. The sole exception I have identified across my extensive experience and research in the field is Michael I. Shamos: he is the sole computer scientist whom I have identified who purports to believe that DREs are acceptably insecure.¹⁰ He is an outlier to the scientific consensus.

I declare under penalty of the perjury laws of the State of Georgia and the United States that the foregoing is true and correct and that this declaration was executed this 17th day of July, 2019 in Princeton, New Jersey.



ANDREW W. APPEL

¹⁰ Dr. Merle S. King, formerly a Professor at Kennesaw State University in Georgia, is also a computer scientist who has defended DREs. In 2004 he published one paper endorsing the use of DREs in Georgia. I cannot find any other scientific articles he has written. Implementing Voting Systems: the Georgia Method, by Brit J. Williams, Merle S. King. Communications of the ACM, October 2004, Vol. 47 No. 10, Pages 39-42.

EXHIBIT C

Ballot-Marking Devices Cannot Ensure the Will of the Voters

Andrew W. Appel, Richard A. DeMillo, and Philip B. Stark

ABSTRACT

The complexity of U.S. elections usually requires computers to count ballots—but computers can be hacked, so election integrity requires a voting system in which paper ballots can be recounted by hand. However, paper ballots provide no assurance unless they accurately record the votes as expressed by the voters.

Voters can express their intent by indelibly hand-marking ballots or using computers called ballot-marking devices (BMDs). Voters can make mistakes in expressing their intent in either technology, but only BMDs are also subject to hacking, bugs, and misconfiguration of the software that prints the marked ballots. Most voters do not review BMD-printed ballots, and those who do often fail to notice when the printed vote is not what they expressed on the touchscreen. Furthermore, there is no action a voter can take to demonstrate to election officials that a BMD altered their expressed votes, nor is there a corrective action that election officials can take if notified by voters—there is no way to deter, contain, or correct computer hacking in BMDs. These are the essential security flaws of BMDs.

Risk-limiting audits can ensure that the votes recorded on paper ballots are tabulated correctly, but no audit can ensure that the votes on paper are the ones expressed by the voter on a touchscreen: Elections conducted on current BMDs cannot be confirmed by audits. We identify two properties of voting systems, *contestability* and *defensibility*, necessary for audits to confirm election outcomes. No available BMD certified by the Election Assistance Commission is contestable or defensible.

Keywords: voting machines, paper ballot, ballot-marking device, election security

INTRODUCTION: CRITERIA FOR VOTING SYSTEMS

ELECTIONS FOR PUBLIC OFFICE and on public questions in the United States or any democracy must produce outcomes based on the votes that voters *express* when they indicate their choices

on a paper ballot or on a machine. Computers have become indispensable to conducting elections, but computers are vulnerable. They can be hacked—compromised by insiders or external adversaries who can replace their software with fraudulent software that deliberately miscounts votes—and they can contain design errors and bugs—hardware or software flaws or configuration errors that result in mis-recording or mis-tabulating votes. Hence there must be some way, *independent* of any software in any computers, to ensure that reported election outcomes are correct, i.e., consistent with the expressed votes as intended by the voters.

Voting systems should be *software independent*, meaning that “an undetected change or error in its software cannot cause an undetectable change or error in an election outcome” (Rivest and Wack

Andrew W. Appel is the Eugene Higgins Professor of Computer Science in the Department of Computer Science at Princeton University in Princeton, New Jersey, USA. Richard A. DeMillo is the Charlotte B. and Roger C. Warren Chair of Computing and a professor in the College of Computing at Georgia Tech in Atlanta, Georgia, USA. Philip B. Stark is a professor in the Department of Statistics at the University of California, Berkeley, in Berkeley, California, USA. The authors are listed alphabetically; they contributed equally to this work.

2006; Rivest 2008; Rivest and Virza 2016). Software independence is similar to tamper-evident packaging: if somebody opens the container and disturbs the contents, it will leave a trace.

The use of software-independent voting systems is supposed to ensure that if someone fraudulently hacks the voting machines to steal votes, we'll know about it. But we also want to know *the true outcome* in order to avoid a do-over election.¹ A voting system is *strongly software independent* if it is software independent and, moreover, a detected change or error in an election outcome (due to change or error in the software) can be corrected using only the ballots and ballot records of the current election (Rivest and Wack 2006; Rivest 2008). Strong software independence combines tamper evidence with a kind of resilience: there's a way to tell whether faulty software caused a problem, and a way to recover from the problem if it did.

Software independence and *strong software independence* are now standard terms in the analysis of voting systems, and it is widely accepted that voting systems should be software independent. Indeed, version 2.0 of the Voluntary Voting System Guidelines (VMSG 2.0) incorporates this principle (U.S. Election Assistance Commission 2017).

But as we will show, these standard definitions are incomplete and inadequate, because the word *undetected* hides several important questions: *Who* detects the change or error in an election outcome? How can a person *prove* that she has detected an error? *What happens* when someone detects an error—does the election outcome remain erroneous? Or conversely: How can an election administrator *prove* that the election outcome not been altered, or prove that the correct outcome was recovered if a software malfunction was detected? The standard definition does not distinguish evidence available to an election official, to the public, or just to a single voter; nor does it consider the possibility of false alarms.

Those questions are not merely academic, as we show with an analysis of ballot-marking devices. Even if some voters “detect” that the printed output is not what they expressed to the ballot-marking device (BMD)—even if some of *those* voters report their detection to election officials—there is no mechanism by which the *election official* can “detect” whether a BMD has been hacked to alter election outcomes. The questions of *who detects*, and *then what happens*, are critical—but unanswered by the standard definitions.

We will define the terms *contestable* and *defensible* to better characterize properties of voting systems that make them acceptable for use in public elections.²

A voting system is *contestable* if an undetected change or error in its software that causes a change or error in an election outcome can always produce *public* evidence that the outcome is untrustworthy. For instance, if a voter selected candidate A on the touchscreen of a BMD, but the BMD prints candidate B on the paper ballot, then this A-vs-B evidence is available to the individual voter, but the voter cannot demonstrate this evidence to anyone else, since nobody else saw—nor should have seen—where the voter touched the screen.³ Thus, the voting system does not provide a way for the voter who observed the misbehavior to prove to anyone else that there was a problem, even if the problems altered the reported outcome. Such a system is therefore not *contestable*.

While the definition of software independence might allow evidence available only to individual voters as “detection,” such evidence does not suffice for a system to be contestable. Contestability is software independence, plus the requirement that “detect” implies “can generate public evidence.” “Trust me” does not count as public evidence. If a voting system is not contestable, then problems voters “detect” might never see the light of day, much less be addressed or corrected.⁴

¹Do-overs are expensive; they may delay the inauguration of an elected official; there is no assurance that the same voters will vote in the do-over election as voted in the original; they decrease public trust. And if the do-over election is conducted with the same voting system that can only detect but not correct errors, then there may need to be a do-over of the do-over, ad infinitum.

²There are other notions connected to contestability and defensibility, although essentially different: Benaloh et al. (2011) define a *P-resilient canvass framework*, *personally verifiable P-resilient canvass framework*, and *privacy-preserving personally verifiable P-resilient canvass frameworks*.

³See footnote 17.

⁴If voters are the only means of detecting and quantifying the effect of those problems—as they are for ballot-marking devices (BMDs)—then in practice the system is not strongly software independent. The reason is that, as we will show, such claims by (some) voters *cannot* correct software-dependent changes to other voters' ballots, and *cannot* be used as the basis to invalidate or correct an election outcome. Thus, BMD-based election systems are not even (weakly) software independent, unless one takes “detection” to mean “somebody claimed there was a problem, with no evidence to support that claim.”

Similarly, while strong software independence demands that a system be able to report the correct outcome even if there was an error or alteration of the software, it does not require *public evidence* that the (reconstructed) reported outcome is correct. We believe, therefore, that voting systems must also be *defensible*. We say that a voting system is defensible if, when the reported electoral outcome is correct, it is possible to generate convincing public evidence that the reported electoral outcome is correct—despite any malfunctions, software errors, or software alterations that might have occurred. If a voting system is not defensible, then it is vulnerable to “crying wolf”: malicious actors could claim that the system malfunctioned when in fact it did not, and election officials will have no way to prove otherwise.

By analogy with *strong software independence*, we define: a voting system is *strongly defensible* if it is defensible and, moreover, a detected change or error in an election outcome (due to change or error in the software) can be corrected (with convincing public evidence) using only the ballots and ballot records of the current election.

In short, a system is contestable if it can generate public evidence of a problem whenever a reported outcome is wrong, while a system is defensible if it can generate public evidence whenever a reported outcome is correct—despite any problems that might have occurred. Contestable systems are publicly tamper-evident; defensible systems are publicly, demonstrably resilient.

Defensibility is a key requirement for *evidence-based elections* (Stark and Wagner 2012): defensibility makes it possible in principle for election officials to generate convincing evidence that the reported winners really won—if the reported winners did really win. (We say an election *system* may be defensible, and an *election* may be evidence-based; there’s much more *process* to an election than just the choice of system.)

Examples

The only known practical technology for contestable, strongly defensible voting is a system of *hand-marked paper ballots*, kept demonstrably physically secure, counted by machine, audited manually, and recountable by hand.⁵ In a hand-marked paper ballot election, ballot-marking software cannot be the source of an error or change-of-election-outcome,

because no software is used in marking ballots. Ballot-scanning-and-counting software can be the source of errors, but such errors can be detected and corrected by audits.

That system is *contestable*: if an optical scan voting machine reports the wrong outcome because it miscounted (because it was hacked, misprogrammed, or miscalibrated), the evidence is *public*: the paper ballots, recounted before witnesses, will not match the claimed results, also witnessed. It is *strongly defensible*: a recount before witnesses can demonstrate that the reported outcome is correct or can find the correct outcome if it was wrong—and provide public evidence that the (reconstructed) outcome is correct. See Section 4, “Contestability/Defensibility of Hand-Marked OpScan,” for a detailed analysis.

Over 40 states now use some form of paper ballot for most voters (Verified Voting Foundation 2018). Most of the remaining states are taking steps to adopt paper ballots. But *not all voting systems that use paper ballots are equally secure*.

Some are not even software independent. Some are software independent but not strongly software independent, contestable, or defensible. In this report we explain:

- *Hand-marked paper ballot* systems are the only practical technology for contestable, strongly defensible voting systems.
- *Some ballot-marking devices* can be software independent, but they not strongly software independent, contestable, or defensible. Hacked or misprogrammed BMDs can alter election outcomes undetectably, so elections conducted using BMDs cannot provide public evidence that reported outcomes are correct. If BMD malfunctions are detected, there is no way to determine who really won. Therefore BMDs should not be used by voters who are able to mark an optical-scan ballot with a pen.
- *All-in-one BMD or DRE+VVPAT voting machines* are not software independent, contestable, or defensible. They should not be used in public elections.

⁵The election must also generate convincing evidence that physical security of the ballots was not compromised, and the audit must generate convincing public evidence that the audit itself was conducted correctly.

BACKGROUND

We briefly review the kinds of election equipment in use, their vulnerability to computer hacking (or programming error), and in what circumstances risk-limiting audits can mitigate that vulnerability.

Voting equipment

Although a voter may form an intention to vote for a candidate or issue days, minutes, or seconds before actually casting a ballot, that intention is a psychological state that cannot be directly observed by anyone else. Others can have access to that intention through what the voter (privately) *expresses* to the voting technology by interacting with it, e.g., by making selections on a BMD or marking a ballot by hand.⁶ Voting systems must accurately record the vote as the voter *expressed* it.

With a *hand-marked paper ballot optical-scan* system, the voter is given a paper ballot on which all choices (candidates) in each contest are listed; next to each candidate is a *target* (typically an oval or other shape) which the voter marks with a pen to indicate a vote. Ballots may be either preprinted or printed (unvoted) at the polling place using *ballot on demand* printers. In either case, the voter creates a tamper-evident record of intent by marking the printed paper ballot with a pen.

Such hand-marked paper ballots may be scanned and tabulated at the polling place using a *precinct-count optical scanner* (PCOS), or may be brought to a central place to be scanned and tabulated by a *central-count optical scanner* (CCOS). Mail-in ballots are typically counted by CCOS machines.

After scanning a ballot, a PCOS machine deposits the ballot in a secure, sealed ballot box for later use in recounts or audits; this is *ballot retention*. Ballots counted by CCOS are also retained for recounts or audits.⁷

Paper ballots can also be hand counted, but in most jurisdictions (especially where there are many contests on the ballot) this is hard to do quickly; Americans expect election-night reporting of unofficial totals. Hand counting—i.e., manually determining votes directly from the paper ballots—is appropriate for audits and recounts.

A *ballot-marking device* provides a computerized user interface (UI) that presents the ballot to voters and captures their expressed selections—for instance, a touchscreen interface or an assistive in-

terface that enables voters with disabilities to vote independently. Voter inputs (expressed votes) are recorded electronically. When a voter indicates that the ballot is complete and ready to be cast, the BMD prints a paper version of the electronically marked ballot. We use the term *BMD* for devices that mark ballots but do not tabulate or retain them, and *all-in-one* for devices that combine ballot marking, tabulation, and retention into the same paper path.

The paper ballot printed by a BMD may be in the same format as an optical-scan form (e.g., with ovals filled as if by hand) or it may list just the names of the candidate(s) selected in each contest. The BMD may also encode these selections into barcodes or QR codes for optical scanning. We discuss issues with barcodes later in this report.

An *all-in-one touchscreen voting machine* combines computerized ballot marking, tabulation, and retention in the same paper path. All-in-one machines come in several configurations:

- DRE+VVPAT machines—direct-recording electronic (DRE) voting machines with a voter-verifiable paper audit trail (VVPAT)—provide the voter a touchscreen (or other) interface, then print a paper ballot that is displayed to the voter under glass. The voter is expected to review this ballot and approve it, after which the machine deposits it into a ballot box. DRE+VVPAT machines do not contain optical scanners; that is, they do not read what is marked on the paper ballot; instead, they tabulate the vote directly from inputs to the touchscreen or other interface.
- BMD+Scanner all-in-one machines⁸ provide the voter a touchscreen (or other) interface to

⁶We recognize that voters make mistakes in expressing their intentions. For example, they may misunderstand the layout of a ballot or express an unintended choice through a perceptual error, inattention, or lapse of memory. The use of touchscreen technology does not necessarily correct for such user errors, as every smartphone user who has mistyped an important text message knows. Poorly designed ballots, poorly designed touchscreen interfaces, and poorly designed assistive interfaces increase the rate of error in voters' expressions of their votes. For the purposes of this report, we assume that properly engineered systems seek to minimize such usability errors.

⁷Regulations and procedures governing custody and physical security of ballots are uneven, and in many cases inadequate, but straightforward to correct because of decades of development of best practices.

⁸Some voting machines, such as the ES&S ExpressVote, can be configured as either a BMD or a BMD+Scanner all-in-one. Others, such as the ExpressVoteXL, work only as all-in-one machines.

input ballot choices and print a paper ballot that is ejected from a slot for the voter to inspect. The voter then reinserts the ballot into the slot, after which the all-in-one BMD+Scanner scans it and deposits it into a ballot box. Or, some BMD+Scanner all-in-one machines display the paper ballot behind plexiglass for the voter to inspect, before mechanically depositing it into a ballot box.

OpSCAN+BMD with separate paper paths. At least one model of voting machine (the Dominion ICP320) contains an optical scanner (opscan) and a BMD in the same cabinet,⁹ so that the optical scanner and BMD-printer are not in the same paper path; no possible configuration of the software could cause a BMD-marked ballot to be deposited in the ballot box without human handling of the ballot. We do not classify this as an *all-in-one* machine.

Hacking

There are many forms of computer hacking. In this analysis of voting machines we focus on the alteration of voting machine software so that it miscounts votes or mis-marks ballots to alter election outcomes. There are many ways to alter the software of a voting machine: a person with physical access to the computer can open it and directly access the memory; one can plug in a special USB thumbdrive that exploits bugs and vulnerabilities in the computer's USB drivers; one can connect to its Wi-Fi port or Bluetooth port or telephone modem (if any) and exploit bugs in those drivers, or in the operating system.

"Air-gapping" a system (i.e., never connecting it to the Internet nor to any other network) does not automatically protect it. Before each election, election administrators must transfer a *ballot definition* into the voting machine by inserting a *ballot definition cartridge* that was programmed on election-administration computers that may have been connected previously to various networks; it has been demonstrated that vote-changing viruses can propagate via these ballot-definition cartridges (Feldman et al. 2007).

Hackers might be corrupt insiders with access to a voting-machine warehouse; corrupt insiders with access to a county's election-administration computers; outsiders who can gain remote access to election-administration computers; outsiders who can gain re-

mote access to voting-machine manufacturers' computers (and "hack" the firmware installed in new machines, or the firmware updates supplied for existing machines), and so on. Supply-chain hacks are also possible: the hardware installed by a voting system vendor may have malware pre-installed by the vendor's component suppliers.¹⁰

Computer systems (including voting machines) have so many layers of software that it is impossible to make them perfectly secure (National Academies of Sciences, Engineering, and Medicine 2018, 89–91). When manufacturers of voting machines use the best known security practices, adversaries may find it more difficult to hack a BMD or optical scanner—but not impossible. Every computer in every critical system is vulnerable to compromise through hacking, insider attacks, or exploiting design flaws.

Election assurance through risk-limiting audits

To ensure that the reported electoral outcome of each contest corresponds to what the voters expressed, the most practical known technology is a *risk-limiting audit* (RLA) of trustworthy paper ballots (Stark 2008; Stark 2009; Lindeman and Stark 2012). The National Academies of Science, Engineering, and Medicine recommend routine RLAs after every election (National Academies of Sciences, Engineering, and Medicine 2018), as do many other organizations and entities concerned with election integrity.¹¹

The *risk limit* of a risk-limiting audit is the maximum chance that the audit will not correct the reported electoral outcome, if the reported outcome is wrong. "Electoral outcome" means the political result—who or what won—not the exact tally. "Wrong" means that the outcome does not correspond to what the voters expressed.

⁹More precisely, the ICP320 optical scanner and the BMD audio+buttons interface are in the same cabinet, but the printer is a separate box.

¹⁰Given that many chips and other components are manufactured in China and elsewhere, this is a serious concern. Carsten Schürmann has found Chinese pop songs on the internal memory of voting machines (C. Schürmann, personal communication, 2018). Presumably those files were left there accidentally—but this shows that malicious code *could* have been pre-installed deliberately, and that neither the vendor's nor the election official's security and quality control measures discovered and removed the extraneous files.

¹¹Among them are the Presidential Commission on Election Administration, the American Statistical Association, the League of Women Voters, and Verified Voting Foundation.

An RLA involves manually inspecting randomly selected paper ballots following a rigorous protocol. The audit stops if and when the sample provides convincing evidence that the reported outcome is correct; otherwise, the audit continues until every ballot has been inspected manually, which reveals the correct electoral outcome if the paper trail is trustworthy. RLAs protect against vote-tabulation errors, whether those errors are caused by failures to follow procedures, misconfiguration, miscalibration, faulty engineering, bugs, or malicious hacking.¹²

The risk limit should be determined as a matter of policy or law. For instance, a 5% risk limit means that, if a reported outcome is wrong solely because of tabulation errors, there is at least a 95% chance that the audit procedure will correct it. Smaller risk limits give higher confidence in election outcomes, but require inspecting more ballots, other things being equal. RLAs never revise a correct outcome.

RLAs can be very efficient, depending in part on how the voting system is designed and how jurisdictions organize their ballots. If the computer results are accurate, an efficient RLA with a risk limit of 5% requires examining just a few—about seven divided by the margin—ballots selected randomly from the contest.¹³ For instance, if the margin of victory is 10% and the results are correct, the RLA would need to examine about $7/10\% = 70$ ballots to confirm the outcome at 5% risk. For a 1% margin, the RLA would need to examine about $7/1\% = 700$ ballots. The sample size does not depend much on the total number of ballots cast in the contest, only on the margin of the winning candidate's victory.

RLAs assume that a full hand tally of the paper trail would reveal the correct electoral outcomes: the paper trail must be trustworthy. Other kinds of audits, such as *compliance audits* (Benaloh et al. 2011; Lindeman and Stark 2012; Stark and Wagner 2012; Stark 2018), are required to establish whether the paper trail itself is trustworthy. Applying an RLA procedure to an untrustworthy paper trail cannot limit the risk that a wrong reported outcome goes uncorrected.

Properly preserved hand-marked paper ballots ensure that expressed votes are identical to recorded votes. But BMDs might not record expressed votes accurately, for instance, if BMD software has bugs, was misconfigured, or was hacked: a BMD printout is not a trustworthy record of the expressed votes. Neither a compliance audit nor an RLA can possibly check whether errors in recording expressed votes

altered election outcomes. RLAs that rely on BMD output therefore cannot limit the risk that an incorrect reported election outcome will go uncorrected.

A paper-based voting system (such as one that uses optical scanners) is systematically more secure than a paperless system (such as DREs) *only if the paper trail is trustworthy and the results are checked against the paper trail using a rigorous method such as an RLA or full manual tally*. If it is possible that error, hacking, bugs, or miscalibration caused the recorded-on-paper votes to differ from the expressed votes, an RLA or even a full hand recount cannot provide convincing public evidence that election outcomes are correct: such a system cannot be *defensible*. In short, paper ballots provide little assurance against hacking if they are never examined or if the paper might not accurately reflect the votes expressed by the voters.

(NON)CONTESTABILITY/ DEFENSIBILITY OF BMDs

A BMD-generated paper trail is not a reliable record of the vote expressed by the voter.

Like any computer, a BMD (or a DRE+VVPAT) is vulnerable to bugs, misconfiguration, hacking, installation of unauthorized (fraudulent) software, and alteration of installed software.

If a hacker sought to steal an election by altering BMD software, what would the hacker program the BMD to do? In cybersecurity practice, we call this the *threat model*.

The simplest threat model is this one: In some contests, not necessarily top-of-the-ticket, change a small percentage of the votes (such as 5%).

In recent national elections, analysts have considered a candidate who received 60% of the vote to have won by a landslide. Many contests are decided by less than a 10% margin. Changing 5% of the votes can change the margin by 10%, because

¹²Risk-limiting audits (RLAs) do not protect against problems that cause BMDs to print something other than what was shown to the voter on the screen, nor do they protect against problems with ballot custody.

¹³Technically, it is the *diluted margin* that enters the calculation. The diluted margin is the number of votes that separate the winner with the fewest votes from the loser with the most votes, divided by the number of ballots cast, including under-votes and invalid votes.

“flipping” a vote for one candidate into a vote for a different candidate changes the difference in their tallies—i.e., the margin—by two votes. If hacking or bugs or misconfiguration could change 5% of the votes, that would be a very significant threat.

Although public and media interests often focus on top-of-the-ticket races such as president and governor, elections for lower offices such as state representatives, who control legislative agendas and redistricting, and county officials, who manage elections and assess taxes, are just as important in our democracy. Altering the outcome of smaller contests requires altering fewer votes, so fewer voters are in a position to notice that their ballots were misprinted. And most voters are not as familiar with the names of the candidates for those offices, so they might be unlikely to notice if their ballots were misprinted, even if they checked.

Research in a real polling place in Tennessee during the 2018 election found that half the voters *didn't look at all* at the paper ballot printed by a BMD, even when they were holding it in their hand and directed to do so while carrying it from the BMD to the optical scanner (DeMillo et al. 2018). Those voters who did look at the BMD-printed ballot spent *an average of 4 seconds* examining it to verify that the eighteen or more choices they made were correctly recorded. That amounts to 222 milliseconds per contest, barely enough time for the human eye to move and refocus under perfect conditions and not nearly enough time for perception, comprehension, and recall (Rayner 2009). A study by other researchers (Bernhard et al. 2020), in a simulated polling place using real BMDs deliberately hacked to alter one vote on each paper ballot, found that only 6.6% of voters told a pollworker something was wrong.^{14,15} The same study found that among voters who examined their hand-marked ballots, half were unable to recall key features of ballots cast moments before, a prerequisite step for being able to recall their own ballot choices. This finding is broadly consistent with studies of effects like “change blindness” or “choice blindness,” in which human subjects fail to notice changes made to choices made only seconds before (Johansson et al. 2008).

Suppose, then, that 10% of voters examine their paper ballots carefully enough to even *see* the candidate's name recorded as their vote for legislator or county commissioner. Of those, perhaps only

half will remember the name of the candidate they intended to vote for.¹⁶

Of those who notice that the vote printed is not the candidate they intended to vote for, what will they think, and what will they do? Will they think, “Oh, I must have made a mistake on the touchscreen,” or will they think, “Hey, the machine is cheating or malfunctioning!” There's no way for the voter to know for sure—voters do make mistakes—and there's *absolutely* no way for the voter to prove to a pollworker or election official that a BMD printed something other than what the voter entered on the screen.^{17,18}

Either way, polling-place procedures generally advise voters to ask a pollworker for a new ballot if theirs does not show what they intended. Pollworkers should void that BMD-printed ballot, and the voter should get another chance to mark a ballot. Anecdotal evidence suggests that many voters are too timid to ask, or don't know that they have the right to ask, or are not sure whom to ask. Even if a voter asks for a new ballot, training for pollworkers is uneven, and we are aware of no formal

¹⁴You might think, “the voter really *should* carefully review their BMD-printed ballot.” But because the scientific evidence shows that voters *do not* (DeMillo et al. 2018) and cognitively *cannot* (Everett 2007) perform this task well, legislators and election administrators should provide a voting system that counts the votes *as voters express them*.

¹⁵Studies of voter confidence about their ability to verify their ballots are not relevant: in typical situations, subjective confidence and objective accuracy are at best weakly correlated. The relationship between confidence and accuracy has been studied in contexts ranging from eyewitness accuracy (Bothwell et al. 1987; Deffenbacher 1980; Wixted and Wells 2017) to confidence in psychological clinical assessments (Desmarais et al. 2010) and social predictions (Dunning et al. 1990). The disconnect is particularly severe at high confidence. Indeed, this is known as “the overconfidence effect.” For a lay discussion, see *Thinking, Fast and Slow* by Nobel economist Daniel Kahnemann (2011).

¹⁶We ask the reader, “do you know the name of the most recent losing candidate for county commissioner?” We recognize that some readers of this document *are* county commissioners, so we ask those readers to imagine the frame of mind of their constituents.

¹⁷You might think, “the voter can prove it by showing someone that the vote on the paper doesn't match the vote onscreen.” But that won't work. On a typical BMD, by the time a paper record is printed and ejected for the voter to hold and examine, the touchscreen no longer shows the voter's choice. You might think, “BMDs should be designed so that the choices still show on the screen for the voter to compare with the paper.” But a hacked BMD could easily alter the on-screen choices to match the paper, *after* the voter hits the “print” button.

¹⁸Voters should *certainly not* video-record themselves voting! That would defeat the privacy of the secret ballot and is illegal in most jurisdictions.

procedure for resolving disputes if a request for a new ballot is refused. Moreover, there is no sensible protocol for ensuring that BMDs that misbehave are investigated—nor can there be, as we argue below.

Let's summarize. If a machine alters votes on 5% of the ballots (enabling it to change the margin by 10%), and 10% of voters check their ballots carefully and 50% of the voters who check notice the error, then optimistically we might expect $5\% \times 10\% \times 50\%$ or 0.25% of the voters to request a new ballot and correct their vote.¹⁹ This means that the machine will change the margin by 9.75% and get away with it.

In this scenario, 0.25% of the voters, one in every 400 voters, has requested a new ballot. You might think, "that's a form of *detection* of the hacking." But it isn't, as a practical matter: a few individual voters may have detected that there was a problem, but there's no procedure by which this translates into any action that election administrators can take to correct the outcome of the election. Polling-place procedures *cannot correct or deter hacking, or even reliably detect it*, as we discuss next. This is essentially the distinction between a system that is merely software independent and one that is contestable: a change to the software that alters the outcome might generate evidence for an alert, conscientious, individual voter, but it does not generate public evidence that an election official can rely on to conclude there is a problem.

Even if some voters notice that BMDs are altering votes, there's no way to correct the election outcome.

That is, BMD voting systems are *not contestable*, *not defensible* (and therefore *not strongly defensible*), and *not strongly software independent*. Suppose a state election official wanted to detect whether the BMDs are cheating, and correct election results, based on actions by those few alert voters who notice the error. What procedures could possibly work against the manipulation we are considering?

1. How about, "If at least 1 in 400 voters claims that the machine misrepresented their vote, void the entire election."²⁰ No responsible authority would implement such a procedure. A few dishonest voters could collaborate to invalidate entire elections simply by falsely claiming that BMDs changed their votes.

2. How about, "If at least 1 in 400 voters claims that the machine misrepresented their vote, then investigate." Investigations are fine, but then what?

The only way an investigation can ensure that the outcome accurately reflects what voters expressed to the BMDs is to void an election in which the BMDs have altered votes and conduct a new election. But how do you know whether the BMDs have altered votes, except based on the claims of the voters?²¹ Furthermore, the investigation itself would suffer from the same problem as above: how can one distinguish between voters who detected BMD hacking or bugs from voters who just want to interfere with an election?

This is the essential security flaw of BMDs: few voters will notice and promptly report discrepancies between what they saw on the screen and what is on the BMD printout, and even when they do notice, there's nothing appropriate that can be done. Even if election officials are convinced that BMDs malfunctioned, *there is no way to determine who really won*.

Therefore, BMDs should not be used by most voters.

Why can't we rely on pre-election and post-election logic and accuracy testing, or parallel testing?

Most, if not all, jurisdictions perform some kind of *logic and accuracy testing* (LAT) of voting equipment before elections. LAT generally involves voting on the equipment using various combinations of selections, then checking whether the equipment tabulated the votes correctly. As the Volkswagen/Audi "Dieselgate" scandal shows, devices can be programmed to behave properly when they are tested but misbehave in use (Contag et al. 2017).

¹⁹This calculation assumes that the 10% of voters who check are in effect a random sample of voters: voters' propensity to check BMD printout is not associated with their political preferences.

²⁰Note that in many jurisdictions, far fewer than 400 voters use a given machine on Election Day: BMDs are typically expected to serve fewer than 300 voters per day. (The vendor ES&S recommended 27,000 BMDs to serve Georgia's 7 million voters, amounting to 260 voters per BMD (Election Systems and Software 2018).) Recall also that the rate one in 400 is tied to the amount of manipulation. What if the malware flipped only one vote in 50, instead of one vote in 20? That could still change the margin by 4%, but—in this hypothetical—would be noticed by only one voter in 1,000, rather than one in 400. The smaller the margin, the less manipulation it would have taken to alter the electoral outcome.

²¹Forensic examination of the BMD might show that it *was* hacked or misconfigured, but it cannot prove that the BMD *was not* hacked or misconfigured.

Therefore, LAT can never prove that voting machines performed properly in practice.

Parallel or “live” testing involves pollworkers or election officials using some BMDs at random times on Election Day to mark (but not cast) ballots with test patterns, then check whether the marks match the patterns. The idea is that the testing is not subject to the “Dieselgate” problem, because the machines cannot “know” they are being tested on Election Day. As a practical matter, the number of tests required to provide a reasonable chance of detecting outcome-changing errors is prohibitive, and even then the system is not *defensible*. See Section 6, “Parallel Testing of BMDs.”

Suppose, counterfactually, that it was practical to perform enough parallel testing to guarantee a large chance of detecting a problem if BMD hacking or malfunction altered electoral outcomes. Suppose, counterfactually, that election officials were required to conduct that amount of parallel testing during every election, and that the required equipment, staffing, infrastructure, and other resources were provided. Even then, the system would not be *strongly defensible*; that is, if testing detected a problem, there would be no way to determine who really won. The only remedy would be a new election.

Don't voters need to check hand-marked ballots, too?

It is always a good idea to check one's work, but there is a substantial body of research (e.g., Reason 2009) suggesting that preventing error as a ballot is being marked is a fundamentally different cognitive task than detecting an error on a previously marked ballot. In cognitively similar tasks, such as proof reading for non-spelling errors, ten percent rates of error detection are common (Reason 2009, 167 et seq.), whereas by carefully attending to the task of correctly marking their ballots, voters apparently can largely avoid marking errors.

A fundamental difference between hand-marked paper ballots and ballot-marking devices is that, with hand-marked paper ballots, voters are responsible for catching and correcting *their own errors*, while if BMDs are used, voters are also responsible for catching *machine errors, bugs, and hacking*. Voters are the *only* people who can detect such problems with BMDs—but, as explained above, if voters do find problems,

there's no way they can prove to poll workers or election officials that there were problems and no way to ensure that election officials take appropriate remedial action.

CONTESTABILITY/DEFENSIBILITY OF HAND-MARKED OPSCAN

The most widely used voting system in the United States is optical-scan counting of hand-marked paper ballots.²² Computers and computer software are used in several stages of the voting process, and if that software is hacked (or erroneous), then the computers will deliberately (or accidentally) report incorrect outcomes.

- Computers are used to prepare the PDF files from which (unvoted) optical-scan ballots are printed, with ovals (or other targets to be marked) next to the names of candidates. Because the optical scanners respond to the *position on the page*, not the name of the candidate nearest the target, computer software could cheat by reordering the candidates on the page.
- The optical-scan voting machine, which scans the ballots and interprets the marks, is driven by computer software. Fraudulent (hacked) software can deliberately record (some fraction of) votes for Candidate A and votes for Candidate B.
- After the voting machine reports the in-the-precinct vote totals (or, in the case of central-count optical scan, the individual-batch vote totals), computers are used to aggregate the various precincts or batches together. Hacked software could cheat in this addition process.

Protection against any or all of these attacks relies on a system of risk-limiting audits, along with compliance audits to check that the chain of custody of ballots and paper records is trustworthy. Without such audits, optical-scan ballots (whether hand marked or machine marked) are neither contestable nor defensible.

²²Verified Voting Foundation, “The Verifier—Polling Place Equipment—November 2020,” *Verified Voting* (2020) <<https://www.verifiedvoting.org/verifier/>> (fetched February 8, 2020).

We analyze the contestability/defensibility of hand-marked optical-scan ballots with respect to each of these threats, assuming a system of RLAs and compliance audits.

- Hacked generation of PDFs leading to fraudulently placed ovals. In this case, a change or error in the computer software *can* change the election outcome: on thousands of ballots, voters place a mark next to the name of candidate A, but (because the candidate name has been fraudulently misplaced on the paper), the (unhacked) optical scanner records this as a vote for candidate B. But an RLA will correct the outcome: a human, inspecting and interpreting this paper ballot, will interpret the mark as a vote for candidate A, as the voter intended. The RLA will, with high probability, conclude that the computer-reported election outcome cannot be confirmed, and a full recount must occur. Thus the system is *contestable*: the RLA produces public evidence that the (computer-reported) outcome is untrustworthy. This full recount (in the presence of witnesses, in view of the public) can provide convincing public evidence of its own correctness; that is, the system is *defensible*.
- Hacked optical-scan vote counter, reporting fraudulent vote totals. In this case, a change or error in the computer software *can* change the election outcome: on thousands of ballots, voters place a mark next to the name of candidate A, but the (hacked) optical scanner records this as a vote for candidate B. But an RLA can detect the incorrect outcome (just as in the case above); the system is *contestable*. And a full recount will produce a correct outcome with public evidence: the system is *defensible*.
- Hacked election-management system (EMS), fraudulently aggregating batches. A risk-limiting audit can detect this problem, and a recount will correct it: the system is contestable and defensible. But actually, contestability and defensibility against this attack is even easier and simpler than RLAs and recounts. Most voting machines (including precinct-count optical scanners) print a “results tape” in the polling place, at the close of the polls (in addition to writing their results electronically to a removable memory card). This results tape is (typically) signed by poll-

workers and by credentialed challengers, and open to inspection by members of the public, before it is transported (with chain-of custody protections) along with the ballot boxes to a secure central location. The county clerk or registrar of voters can (and in many counties, does) inspect these paper records to verify that they correspond to the precinct-by-precinct machine-reported aggregation. Errors (or fraud) in aggregation can be detected and corrected without the need to inspect individual ballots: the system is contestable and defensible against this class of errors.

END-TO-END VERIFIABLE (E2E-V) SYSTEMS

In all BMD systems currently on the market, and in all BMD systems certified by the Election Assistance Commission (EAC), the printed ballot or ballot summary is the only channel by which voters can verify the correct recording of their ballots, independently of the computers. The analysis in this article applies to all of those BMD systems.

There is a class of voting systems called “end-to-end verifiable” (E2E-V), which provide an alternate mechanism for voters to verify their votes (Benaloh et al. 2014; Appel 2018b). The basic idea of an E2E-V system is that a cryptographic protocol encodes the vote; mathematical properties of the cryptographic system allow the voters to verify (probabilistically) that their vote has been accurately counted, but does not compromise the secret ballot by allowing voters to prove how they voted. E2E-V systems have not been adopted in public elections (except that Scantegrity was used for municipal elections in Takoma Park, Maryland, in 2009 and 2011).

Each E2E-V system requires its own analysis of contestability/defensibility.

Scantegrity (Chaum et al. 2008) is a system of preprinted optical-scan ballots, counted by conventional precinct-count optical scanners, but with an additional security feature: when the voter fills in an oval with a special pen, the oval is mostly darkened (so it's counted conventionally by the optical scanner), but two-letter code is also revealed that the voter can (optionally) use in the cryptographic protocol. *Scantegrity* is contestable/defensible, but not because of its E2E-V properties: since it's an add-on to a conventional optical-scan system

with hand-marked paper ballots, RLAs and compliance audits can render this system contestable/defensible.

Prêt-à-Voter (Ryan et al. 2009) is the system in which the voter separates the candidate list from the oval-target list after marking the ballot and before deposit into the optical scanner. This system can be made contestable, with difficulty: the auditing procedure requires participation of the voters in an unintuitive cryptographic challenge. It is not clear that the system is defensible: if this cryptographic challenge proves that the blank ballots have been tampered with, then no recount can reliably reconstruct the true result with public evidence.

STAR-Vote (Benaloh et al. 2013) is a DRE+VV-PAT system with a smart ballot box. Voters interact with a device that captures their votes electronically and prints a paper record that voters can inspect, but the electronic votes are held “in limbo” until the paper ballot is deposited in the smart ballot box. The ballot box does not read the votes from the ballot; rather, depositing the ballot tells the system that it has permission to cast the votes it had already recorded from the touchscreen. The claimed advantage of *STAR-Vote* (and other systems that use the “Benaloh challenge”) is that RLAs and ballot-box chain-of-custody are not required in order to obtain software independence. To ensure that the E2E-V cryptographic protocol has correctly recorded each vote, the voter can “challenge” the system to prove that the cryptographic encoding of the ballot records the vote actually printed on the paper ballot. To do so, the voter must discard (void) this ballot and vote a fresh ballot; this is because the challenge process reveals the vote to the public, and a voting system must preserve the secrecy of the (cast) ballots. Thus, the voter cannot ensure the correct encoding of their true ballot, but (since *STAR-Vote* must print the ballot before knowing whether the voter will challenge), the voter can ensure it with any desired *error probability*.

STAR-Vote is software independent but it is not contestable or defensible. The reason is that, while the challenge can produce public evidence that a machine did not accurately encrypt the plaintext vote on the ballot, if the machine prints the wrong plaintext vote and a correct encryption of that incorrect vote, there is no evidence the voter can use to prove that to anyone else.

No E2E-V system is currently certified by the EAC, nor to our knowledge is any such system

under review for certification, nor are any of the five major voting-machine vendors offering such a system for sale.²³

PARALLEL TESTING OF BMDs

Wallach (2019) has proposed (in response to earlier drafts of this article) that contestability/defensibility failure of BMDs could be mitigated by *parallel testing*, which he also calls “live auditing.” Stark (2019) has analyzed Wallach’s proposal in detail. Here we provide a summary of the proposal and the analyses.

One might like to test each BMD before the election to make sure it’s not hacked. Unfortunately, since the computer in a voting machine (including BMDs) has a real-time clock, the software (including fraudulent vote-stealing software) knows whether it’s Election Day or not. Fraudulent software can make sure not to cheat except on Election Day.

The idea of parallel testing is to have trained auditors test the BMDs, at random times during an actual election: use the BMD to prepare a ballot, inspect that ballot to ensure it’s marked correctly, then discard the ballot. The same BMDs in use during the polling will be selected, from time to time, for such test, right there in the polling places.

If the BMDs cheat with uniform random probability p , and if the BMD cannot distinguish an auditor from an ordinary voter, then after n random audits the probability of detecting the malware is $1 - (1 - p)^n$. If $p = 5\%$ and $n = 240$, then the probability of detection is 91%.

Unfortunately, the attacker is not constrained to cheat with uniform random probability; or, to put it another way, BMD malware may indeed be able to distinguish auditors from ordinary voters. Stark (2019) discusses many ways in which the “signature” of how auditors interact with the BMD may differ from ordinary voters, enough to give clues

²³Some vendors, notably Scytl, have sold systems advertised as E2E-V in other countries. Those systems were not in fact E2E-V. Moreover, serious security flaws have been found in their implementations. See, e.g., S.J. Lewis, O. Pereira, and V. Teague, “Ceci N’est Pas une Preuve: The Use of Trapdoor Commitments in Bayer-Groth Proofs and the Implications for the Verifiability of the Scytl-SwissPost Internet Voting System” (March 12, 2019), <<https://people.eng.unimelb.edu.au/vjteague/UniversalVerifiabilitySwissPost.pdf>>.

to the malware about whether to cheat.²⁴ Therefore, one cannot simply multiply $(1 - p)^n$ and calculate a probability of detection.

While auditors might try to build an accurate model of voter behavior for live audits, that approach is doomed by privacy concerns and by the “curse of dimensionality”: election officials would have to record every nuance of voter behavior (preferences across contests; language settings, font settings, and other UI settings; timing, including speed of voting and hesitation; on-screen review; etc.) for millions of voters to accurately approximate voter behavior.

There are many logistical problems with “live auditing.” It would require additional voting machines (because testing requires additional capacity), staff, infrastructure, and other resources, *on Election Day* when professional staff is most stretched. One must be prepared to perform the audits at the busiest times of day; even that will cause lines of voters to lengthen, because otherwise the malware can simply cheat only at the busy times. Live auditing must be done in view of the voters (one cannot carry the voting machine into another room to do it), but some election officials are concerned that the creation of test ballots in the polling place could be perceived as a threat of ballot-box stuffing.

No state, to our knowledge, has implemented parallel testing or live auditing of BMDs.

In any case, we can assess the contestability and defensibility of parallel testing.

With a sufficiently high rate of parallel testing, and a sufficiently sophisticated randomization of auditor behavior, it may be possible to make BMDs with parallel testing *contestable*: an audit could detect *and prove* mismarking of paper ballots.

But BMDs with parallel testing is not *defensible*. It will be extremely difficult for an election official to generate convincing public evidence that the audit *would have* detected mismarking, if mismarking were occurring. To generate that public evidence, the election official would have to reveal substantial detail about the parallel-testing protocol: how, exactly, the random selection of times to test is made; how, exactly, the random selection is made of what candidates to vote for in the tests. Revealing such details of the protocol allows the attacker to analyze the protocol for clues about how and when to cheat with less chance of detection.

Furthermore, parallel testing has a severe disadvantage in comparison with other contestable/defensible paper-ballot-based voting systems: If

the auditors detect that the BMDs have mismarked a ballot—even once—the entire election must be invalidated, and a do-over election must be held. This is because the auditor will have detected evidence that the BMDs in this election have been systematically mismarking ballots for some proportion of *all* voters. No recount of the paper ballots can correct this.

In contrast, if optical scanners are hacked to cheat on hand-marked paper ballots, the correct outcome can be calculated by a full hand recount of the paper ballots.²⁵

Wallach also suggests, instead of parallel testing, the use of spoiled-ballot rates as a measure of BMD cheating. Suppose, when BMDs are not cheating, the baseline rate of spoiled ballots (i.e., voters asking for a “do-over” of their BMD marked ballot) is 1%. Suppose the machines are cheating on 5% of the ballots, and 6% of voters notice this, and ask for a do-over. Then the spoiled ballot rate increases to 1.3%. The election administrator is supposed to act upon this discrepancy. But the only meaningful action the administrator could take is to invalidate the entire election, and call for a do-over election. This is impractical.

Moreover, the underlying “natural” rate of spoilage will not be known exactly, and will vary from election to election, even if the machines function flawlessly. The natural rate might depend on the number of contests on the ballot, the complexity of voting rules (e.g., instant-runoff voting [IRV] versus plurality), ballot layout, and many other factors. For any rule, there will be a tradeoff between false alarms and failures to detect problems.

To continue the previous hypothetical, suppose that spoiled ballots follow a Poisson distribution (there is no reason to think that they do). Imagine that the theoretical rate is known to be 1% if the

²⁴For example, BMDs do “know” their own settings and other aspects of each voting session, so malware can use that information to target sessions that use the audio interface, increase the font size, use the sip-and-puff interface, set the language to something other than English, or take much longer than average to vote. (Voters who use those settings might be less likely to be believed if they report that the equipment altered their votes.) For parallel testing to have a good chance of detecting all outcome-changing problems, the tests must have a large chance of probing *every* combination of settings and voting patterns that includes enough ballots to change any contest result. It is not practical.

²⁵Provided, of course, that secure chain of custody of the ballot boxes can be demonstrated.

BMDs function correctly, and known to be 1.3% if the BMDs malfunction. How many votes must be cast for it to be possible to limit the chance of a false alarm to 1%, while ensuring a 99% chance of detecting a real problem? The answer is 28,300 votes. If turnout is roughly 50%, jurisdictions (or contests) with fewer than 60,000 voters could not in principle limit the chance of false positives and of false negatives to 1%—even under these optimistic assumptions and simplifications. Twenty-three of California's 58 counties have fewer than 60,000 registered voters.

OTHER TRADEOFFS, BMDs VERSUS HAND-MARKED OPSCAN

Supporters of ballot-marking devices advance several other arguments for their use.

Mark legibility. A common argument is that a properly functioning BMD will generate clean, error-free, unambiguous marks, while hand-marked paper ballots may contain mistakes and stray marks that make it impossible to discern a voter's intent. However appealing this argument seems at first blush, the data are not nearly so compelling. Experience with statewide recounts in Minnesota and elsewhere suggest that truly ambiguous handmade marks are very rare.²⁶ For instance, 2.9 million hand-marked ballots were cast in the 2008 Minnesota race between Al Franken and Norm Coleman for the U.S. Senate. In a manual recount, between 99.95% and 99.99% of ballots were unambiguously marked.^{27,28} In addition, usability studies of hand-marked bubble ballots—the kind in most common use in U.S. elections—indicate a *voter* error rate of 0.6%, much lower than the 2.5%–3.7% error rate for machine-marked ballots (Everett 2007).²⁹ Thus, mark legibility is not a good reason to adopt BMDs for all voters.

Undervotes, overvotes. Another argument offered for BMDs is that the machines can alert voters to undervotes and prevent overvotes. That is true, but modern PCOS systems can also alert a voter to overvotes and undervotes, allowing a voter to eject the ballot and correct it.

Bad ballot design. Ill-designed paper ballots, just like ill-designed touchscreen interfaces, may lead to unintentional undervotes (Norden et al. 2008). For instance, the 2006 Sarasota, Florida, touchscreen ballot was badly designed. The 2018 Broward County, Flor-

ida, opscan ballot was badly designed: it violated three separate guidelines from the EAC's 2007 publication, "Effective Designs for the Administration of Federal Elections, Section 3: Optical Scan Ballots" (U.S. Election Assistance Commission 2007). In both of these cases (touchscreens in 2006, hand-marked optical-scan in 2018), undervote rates were high. The solution is to follow standard, published ballot-design guidelines and other best practices, both for touchscreens and for hand-marked ballots (Appel 2018c; Norden et al. 2008).

Low-tech paper-ballot fraud. All paper ballots, however they are marked, are vulnerable to *loss*, *ballot-box stuffing*, *alteration*, and *substitution* between the time they are cast and the time they are recounted. That's why it is so important to make sure that ballot boxes are always in multiple-person (preferably bipartisan) custody whenever they are handled, and that appropriate physical security measures are in place. Strong, verifiable chain-of-custody protections are essential.

Hand-marked paper ballots are vulnerable to alteration by anyone with a pen. Both hand-marked and BMD-marked paper ballots are vulnerable to substitution: anyone who has poorly supervised access to a legitimate BMD during election day can create fraudulent ballots, not necessarily to deposit them in the ballot box immediately (in case the

²⁶States do need clear and complete regulations for interpreting voter marks.

²⁷During the recount, the Coleman and Franken campaigns initially challenged a total of 6,655 ballot-interpretation decisions made by the human recounters. The State Canvassing Board asked the campaigns to voluntarily withdraw all but their most serious challenges, and in the end approximately 1,325 challenges remained. That is, approximately 5 ballots in 10,000 were ambiguous enough that one side or the other felt like arguing about it. The State Canvassing Board, in the end, classified all but 248 of these ballots as votes for one candidate or another. That is, approximately 1 ballot in 10,000 was ambiguous enough that the bipartisan recount board could not determine an intent to vote." (Appel 2009; see also Office of the Minnesota Secretary of State 2009).

²⁸We have found that some local election officials consider marks to be ambiguous if *machines* cannot read the marks. That is a different issue from *humans* being unable to interpret the marks. Errors in machine interpretation of voter intent can be dealt with by manual audits: if the reported outcome is wrong because machines misinterpreted handmade marks, an RLA has a known, large chance of correcting the outcome.

²⁹Better designed user interfaces (UI) might reduce the error rate for machine-marked ballots below the historical rate for direct-recording electronic (DRE) voting machines; however, UI improvements cannot keep BMDs from printing something other than what the voter is shown on the screen.

ballot box is well supervised on Election Day) but with the hope of substituting it later in the chain of custody.³⁰

All those attacks (on hand-marked and on BMD-marked paper ballots) are fairly low-tech. There are also higher-tech ways of producing ballots indistinguishable from BMD-marked ballots for substitution into the ballot box if there is inadequate chain-of-custody protection.

Accessible voting technology. When hand-marked paper ballots are used with PCOS, there is (as required by law) also an accessible voting technology available in the polling place for voters unable to mark a paper ballot with a pen. This is typically a BMD or a DRE. When the accessible voting technology is not the same as what most voters vote on—when it is used by very few voters—it may happen that the accessible technology is ill-maintained or even (in some polling places) not even properly set up by pollworkers. This is a real problem. One proposed solution is to require all voters to use the same BMD or all-in-one technology. But the failure of some election officials to properly maintain their accessible equipment is not a good reason to adopt BMDs for *all* voters. Among other things, it would expose all voters to the security flaws described above.³¹ Other advocates object to the idea that disabled voters must use a different method of marking ballots, arguing that their rights are thereby violated. Both the Help America Vote Act (HAVA) and the Americans with Disabilities Act (ADA) require reasonable accommodations for voters with physical and cognitive impairments, but neither law requires that those accommodations must be used by all voters. To best enable and facilitate participation by all voters, each voter should be provided with a means of casting a vote best suited to their abilities.

Ballot printing costs. Preprinted optical-scan ballots cost 20–50 cents each.³² Blank cards for BMDs cost up to 15 cents each, depending on the make and model of BMD.³³ But optical-scan ballots must be preprinted for as many voters as *might* show up, whereas blank BMD cards are consumed in proportion to how many voters *do* show up. The Open Source Election Technology Institute (OSET) conducted an independent study of total life cycle costs³⁴ for hand-marked paper ballots and BMDs in conjunction with the 2019 Georgia legislative debate regarding BMDs (Perez 2019). OSET concluded that, even in the most optimistic (i.e., lowest cost) scenario for BMDs and the most pessimistic (i.e., highest cost)

scenario for hand-marked paper ballots and ballot-on-demand (BOD) printers—which can print unmarked ballots as needed—the total lifecycle costs for BMDs would be higher than the corresponding costs for hand-marked paper ballots.³⁵

Vote centers. To run a vote center that serves many election districts with different ballot styles, one must be able to provide each voter a ballot containing the contests that voter is eligible to vote in, possibly in a number of different languages. This is easy with BMDs, which can be programmed with all the appropriate ballot definitions. With preprinted optical-scan ballots, the PCOS can be programmed to *accept* many different ballot styles, but the vote center must still maintain *inventory* of many different ballots. BOD printers are another economical alternative for vote centers.³⁶

Paper/storage. BMDs that print summary cards rather than full-face ballots can save paper and storage space. However, many BMDs print full-face ballots—so they do not save storage—while many

³⁰Some BMDs print a barcode indicating when and where the ballot was produced, but that does not prevent such a substitution attack against currently Election Assistance Commission (EAC)-certified, commercially available BMDs. We understand that systems under development might make ballot-substitution attacks against BMDs more difficult.

³¹Also, some accessibility advocates argue that requiring disabled voters to use BMDs compromises their privacy since hand-marked ballots are easily distinguishable from machine marked ballots. That issue can be addressed without BMDs—for all: Accessible BMDs are already available and in use that mark ballots with marks that cannot easily be distinguished from hand-marked ballots.

³²Single-sheet (one- or two-side) ballots cost 20–28 cents; double-sheet ballots needed for elections with many contests cost up to 50 cents.

³³Ballot cards for ES&S ExpressVote cost about 15 cents. New Hampshire's (One4All/Prime III) BMDs used by sight-impaired voters use plain paper that is less expensive.

³⁴They include not only the cost of acquiring and implementing systems but also the ongoing licensing, logistics, and operating (purchasing paper stock, printing, and inventory management) costs.

³⁵Ballot-on-demand (BOD) printers currently on the market arguably are best suited for vote centers, but less expensive options suited for polling places could be developed. Indeed, BMDs that print full-face ballots could be re-purposed as BOD printers for polling place use, with modest changes to the programming.

³⁶Ballot-on-demand printers *may* require maintenance such as replacement of toner cartridges. This is readily accomplished at a vote center with a professional staff. Ballot-on-demand printers may be a less attractive option for many small precincts on Election Day, where there is no professional staff—but on the other hand, they are less necessary, since far fewer ballot styles will be needed in any one precinct.

BMDs that print summary cards (which could save storage) use thermal printers and paper that is fragile and can fade in a few months.³⁷

Advocates of hand-marked paper ballot systems advance these additional arguments.

Cost. Using BMDs for all voters substantially increases the cost of acquiring, configuring, and maintaining the voting system. One PCOS can serve 1,200 voters in a day, while one BMD can serve only about 260 (Election Systems and Software 2018)—though both these numbers vary greatly depending on the length of the ballot and the length of the day. OSET analyzed the relative costs of acquiring BMDs for Georgia’s nearly seven million registered voters versus a system of hand-marked paper ballots, scanners, and BOD printers (Perez 2019). A BMD solution for Georgia would cost taxpayers between three and five times more than a system based on hand-marked paper ballots. Open-source systems might eventually shift the economics, but current commercial universal-use BMD systems are more expensive than systems that use hand-marked paper ballots for most voters.

Mechanical reliability and capacity. Pens are likely to have less downtime than BMDs. It is easy and inexpensive to get more pens and privacy screens when additional capacity is needed. If a precinct-count scanner goes down, people can still mark ballots with a pen; if the BMD goes down, voting stops. Thermal printers used in DREs with VVPAT are prone to jams; those in BMDs might have similar flaws.

These secondary pros and cons of BMDs do not outweigh the primary security and accuracy concern: BMDs, if hacked or erroneously programmed, can change votes in a way that is not correctable. BMD voting systems are not contestable or defensible. Audits that rely on BMD printout cannot make up for this defect in the paper trail: they cannot reliably detect or correct problems that altered election outcomes.

Barcodes

A controversial feature of some BMDs allows them to print one-dimensional or two-dimensional barcodes on the paper ballots. A one-dimensional barcode resembles the pattern of vertical lines used to identify products by their universal product codes. A two-dimensional barcode or QR code is a rectangular area covered in coded image *modules*

that encode more complex patterns and information. BMDs print barcodes on the same paper ballot that contains human-readable ballot choices. Voters using BMDs are expected to verify the human-readable printing on the paper ballot card, but the presence of barcodes with human-readable text poses some significant problems.

Barcodes are not human readable. The whole purpose of a paper ballot is to be able to recount (or audit) the voters’ votes in a way independent of any (possibly hacked or buggy) computers. If the official vote on the ballot card is the barcode, then it is impossible for the voters to verify that the official vote they cast is the vote they expressed. Therefore, before a state even *considers* using BMDs that print barcodes (and we do not recommend doing so), the state must ensure by statute that recounts and audits are based *only* on the human-readable portion of the paper ballot. Even so, audits based on untrustworthy paper trails suffer from the verifiability the problems outlined above.

Ballot cards with barcodes contain two different votes. Suppose a state does ensure by statute that recounts and audits are based on the human-readable portion of the paper ballot. Now a BMD-marked ballot card with both barcodes and human-readable text contains two different votes in each contest: the barcode (used for electronic tabulation), and the human-readable selection printout (official for audits and recounts). In few (if any) states has there even been a discussion of the legal issues raised when the official markings to be counted differ between the original count and a recount.

Barcodes pose technical risks. Any coded input into a computer system—including wired network packets, Wi-Fi, USB thumbdrives, *and barcodes*—pose the risk that the input-processing software can be vulnerable to attack via deliberately ill-formed input. Over the past two decades, many such vulnerabilities have been documented on *each* of these channels (including barcode readers) that, in the worst case,

³⁷The California Top-To-Bottom Review (TTBR) of voting systems found that thermal paper can also be covertly spoiled wholesale using common household chemicals. <<https://votingsystems.cdn.sos.ca.gov/oversight/ttbr/red-diebold.pdf>> (last visited April 8, 2019; Matt Bishop, Principal Investigator). The fact that thermal paper printing can fade or deteriorate rapidly might mean it does not satisfy the federal requirement to preserve voting materials for 22 months (U.S. Code Title 52, Chapter 207, Sec. 20701, as of April 2020).

give the attacker complete control of a system.³⁸ If an attacker were able to compromise a BMD, the barcodes are an attack vector for the attacker to take over an optical scanner (PCOS or CCOS), too. Since it is good practice to close down all such unneeded attack vectors into PCOS or CCOS voting machines (e.g., don't connect your PCOS to the Internet!), it is also good practice to avoid unnecessary attack channels such as barcodes.

INSECURITY OF ALL-IN-ONE BMDs

Some voting machines incorporate a BMD interface, printer, and optical scanner into the same cabinet. Other DRE+VVPAT voting machines incorporate ballot-marking, tabulation, and paper-printout retention, but without scanning. These are often called "all-in-one" voting machines. To use an all-in-one machine, the voter makes choices on a touchscreen or through a different accessible interface. When the selections are complete, the BMD prints the completed ballot for the voter to review and verify, before depositing the ballot in a ballot box attached to the machine.

Such machines are especially unsafe: like any BMD described in Section 3, "(Non)Contestability/Defensibility of BMDs," they are not contestable or defensible, but in addition, if hacked they can print votes onto the ballot *after* the voter last inspects the ballot.

- The ES&S ExpressVote (in all-in-one mode) allows the voter to mark a ballot by touchscreen or audio interface, then prints a paper ballot card and ejects it from a slot. The voter has the opportunity to review the ballot, then the voter redeposits the ballot into the same slot, where it is scanned and deposited into a ballot box.
- The ES&S ExpressVoteXL allows the voter to mark a ballot by touchscreen or audio interface, then prints a paper ballot and displays it under glass. The voter has the opportunity to review the ballot, then the voter touches the screen to indicate "OK," and the machine pulls paper ballot up (still under glass) and into the integrated ballot box.
- The Dominion ImageCast Evolution (ICE) allows the voter to deposit a hand-marked paper ballot, which it scans and drops into the attached ballot box. *Or*, a voter can use a touchscreen or audio interface to direct the marking of a paper

ballot, which the voting machine ejects through a slot for review; then the voter redeposits the ballot into the slot, where it is scanned and dropped into the ballot box.

In all three of these machines, the ballot-marking printer is in the same paper path as the mechanism to deposit marked ballots into an attached ballot box. This opens up a very serious security vulnerability: the voting machine can mark the paper ballot (to add votes or spoil already-cast votes) after the last time the voter sees the paper, and then deposit that marked ballot into the ballot box without the possibility of detection.

Vote-stealing software could easily be constructed that looks for *undervotes* on the ballot, and marks those unvoted spaces for the candidate of the hacker's choice. This is very straightforward to do on optical-scan bubble ballots (as on the Dominion ICE) where undervotes are indicated by no mark at all. On machines such as the ExpressVote and ExpressVoteXL, the normal software indicates an undervote with the words "no selection made" on the ballot summary card. Hacked software could simply leave a blank space there (most voters wouldn't notice the difference), and then fill in that space and add a matching bar code after the voter has clicked "cast this ballot."

An even worse feature of the ES&S ExpressVote and the Dominion ICE is the *auto-cast* configuration setting (in the manufacturer's standard software) that allows the voter to indicate, "don't eject the ballot for my review, just print it and cast it without me looking at it." If fraudulent software were installed in the ExpressVote, it could change *all* the votes of any voter who selected this option, because the voting machine software would know *in advance of printing* that the voter had waived the opportunity to inspect the printed ballot. We call this auto-cast feature "permission to cheat" (Appel 2018a).

Regarding these all-in-one machines, we conclude:

³⁸An example of a barcode attack is based on the fact that many commercial barcode-scanner components (which system integrators use to build cash registers or voting machines) treat the barcode scanner using the same operating-system interface as if it were a keyboard device; and then some operating systems allow "keyboard escapes" or "keyboard function keys" to perform unexpected operations.

- Any machine with ballot printing in the same paper path with ballot deposit is not *software independent*; it is not the case that “an error or fault in the voting system software or hardware cannot cause an undetectable change in election results.” Therefore such all-in-one machines do not comply with the VVSG 2.0 (the Election Assistance Commission’s Voluntary Voting Systems Guidelines). Such machines are not contestable or defensible, either.
- All-in-one machines on which all voters use the BMD interface to mark their ballots (such as the ExpressVote and ExpressVoteXL) *also* suffer from the same serious problem as ordinary BMDs: most voters do not review their ballots effectively, and elections on these machines are not contestable or defensible.
- The auto-cast option for a voter to allow the paper ballot to be cast without human inspection is particularly dangerous, and states must insist that vendors disable or eliminate this mode from the software. However, even disabling the auto-cast feature does not eliminate the risk of undetected vote manipulation.

Remark

The Dominion ImageCast Precinct ICP320 is a precinct-count optical scanner (PCOS) that also contains an audio+buttons ballot-marking interface for disabled voters. This machine can be configured to cast electronic-only ballots from the BMD interface, or an external printer can be attached to print paper optical-scan ballots from the BMD interface. When the external printer is used, that printer’s paper path is *not* connected to the scanner+ballot-box paper path (a person must take the ballot from the printer and deposit it into the scanner slot). Therefore this machine is as safe to use as any PCOS with a separate external BMD.

CONCLUSION

Ballot-marking devices produce ballots that do not necessarily record the vote expressed by the voter when they enter their selections on the touchscreen: hacking, bugs, and configuration errors can cause the BMDs to print votes that differ from what the voter entered and verified electroni-

cally. Because outcome-changing errors in BMD printout do not produce public evidence, BMD systems are not *contestable*. Because there is no way to generate convincing public evidence that reported outcomes are correct despite any BMD malfunctions that might have occurred, BMD systems are not *defensible*. Therefore, BMDs should not be used by voters who can hand mark paper ballots.

All-in-one voting machines, which combine ballot-marking and ballot-box-deposit into the same paper path, are even worse. They have all the disadvantages of BMDs (they are not contestable or defensible), and they can mark the ballot after the voter has inspected it. Therefore they are not even *software independent*, and should not be used by those voters who are capable of marking, handling, and visually inspecting a paper ballot.

When computers are used to record votes, the original transaction (the voter’s expression of the votes) is not documented in a verifiable way.³⁹ When pen and paper are used to record the vote, the original expression of the vote *is* documented in a verifiable way (if demonstrably secure chain of custody of the paper ballots is maintained). Audits of elections conducted with hand-marked paper ballots, counted by optical scanners, can ensure that reported election outcomes are correct. Audits of elections conducted with BMDs *cannot* ensure that reported outcomes are correct.

REFERENCES

- Appel, A.W. 2009. “Optical-Scan Voting Extremely Accurate in Minnesota.” *Freedom to Tinker*. January. <<https://freedom-to-tinker.com/2009/01/21/optical-scan-voting-extremely-accurate-minnesota/>>.
- Appel, A.W. 2018a. “Serious Design Flaw in ESS ExpressVote Touch-Screen: “Permission to Cheat.”” *Freedom to Tinker*. September. <<https://freedom-to-tinker.com/2018/09/14/serious-design-flaw-in-ess-expressvote-touchscreen-permission-to-cheat/>>.

³⁹It is conceivable that cryptographic protocols like those used in E2E-V systems could be used to create BMD-based systems that are contestable and defensible, but no such system exists, nor, to our knowledge, has such a design been worked out in principle. Existing E2E-V systems that use a computer to print (encrypted) selections are neither contestable nor defensible, as explained in Section 1, “Introduction: Criteria for Voting Systems.”

- Appel, A.W. 2018b. "End-to-End Verifiable Elections." *Freedom to Tinker*. November. <<https://freedom-to-tinker.com/2018/11/05/end-to-end-verifiable-elections/>>.
- Appel, A.W. 2018c. "Florida Is the Florida of Ballot-Design Mistakes." *Freedom to Tinker*. November 2018. <<https://freedom-to-tinker.com/2018/11/14/florida-is-the-florida-of-ballot-design-mistakes/>>.
- Bell, S., J. Benaloh, M.D. Byrne, D. DeBeauvoir, B. Eakin, G. Fisher, P. Kortum, N. McBurnett, J. Montoya, M. Parker, O. Pereira, P.B. Stark, D.S. Wallach, and M. Winn. 2013. "STAR-Vote: A Secure, Transparent, Auditable, and Reliable Voting System." *USENIX Journal of Election Technology and Systems (JETS)* 1(1): 18–37.
- Benaloh, J., D. Jones, E. Lazarus, M. Lindeman, and P.B. Stark. 2011. "SOBA: Secrecy-Preserving Observable Ballot-Level Audits." In *Proceedings of the 2011 Electronic Voting Technology Workshop / Workshop on Trustworthy Elections (EVT/WOTE '11)*. USENIX.
- Benaloh, Josh, Ronald L. Rivest, Peter Y.A. Ryan, Philip B. Stark, Vanessa Teague, and Poorvi L. Vora. 2014. "End-to-End Verifiability." *ArXiv.org*. February 2. <<https://arxiv.org/ftp/arxiv/papers/1504/1504.03778.pdf>>.
- Bernhard, Matthew, Allison McDonald, Henry Meng, Jensen Hwa, Nakul Bajaj, Kevin Chang, and J. Alex Halderman. 2020. "Can Voters Detect Malicious Manipulation of Ballot Marking Devices?" In *41st IEEE Symposium on Security and Privacy*. IEEE.
- Bothwell, R.K., K.A. Deffenbacher, and J.C. Brigham. 1987. "Correlation of Eyewitness Accuracy and Confidence: Optimality Hypothesis Revisited." *Journal of Applied Psychology* 72(4): 691–695.
- Chaum, D., A. Essex, R.T. Carback III, J. Clark, S. Popoveniuc, A.T. Sherman, and P. Vora. 2008. "Scantegrity: End-to-End Voter Verifiable Optical-Scan Voting." *IEEE Security & Privacy* 6(3): 40–46.
- Contag, Moritz, Guo Li, Andre Pawlowski, Felix Domke, Kirill Levchenko, Thorsten Holz, and Stefan Savage. 2017. "How They Did It: An Analysis of Emission Defeat Devices in Modern Automobiles." In *2017 IEEE Symposium on Security and Privacy*, 231–250. San Jose, CA: IEEE.
- Deffenbacher, K. 1980. "Eyewitness Accuracy and Confidence: Can We Infer Anything About Their Relation?" *Law and Human Behavior* 4: 243–260.
- DeMillo, R., R. Kadel, and M. Marks. 2018. "What Voters Are Asked to Verify Affects Ballot Verification: A Quantitative Analysis of Voters' Memories of Their Ballots." *SSRN*. November. <<https://ssrn.com/abstract=3292208>>.
- Desmarais, S.L., T.L. Nicholls, J. D. Read, and J. Brink. 2010. "Confidence and Accuracy in Assessments of Short-Term Risks Presented by Forensic Psychiatric Patients." *Journal of Forensic Psychiatry & Psychology* 21(1): 1–22.
- Dunning, D., D.W. Griffin, J.D. Milojkovic, and L. Ross. 1990. "The Overconfidence Effect in Social Prediction." *Journal of Personality and Social Psychology* 58: 568–581.
- Election Systems and Software. 2018. *State of Georgia Electronic Request for Information New Voting System*. Event Number: 47800-SOS0000035. August 24. <<http://sos.ga.gov/admin/files/ESS%20RFI%20-%20Final%20-%20Redacted.pdf>>.
- Everett, S.P. 2007. "The Usability of Electronic Voting Machines and How Votes Can Be Changed Without Detection." PhD thesis. Rice University.
- Feldman, A.J., J.A. Halderman, and E.W. Felten. 2007. "Security Analysis of the Diebold AccuVote-TS Voting Machine." In *2007 USENIX/ACCURATE Electronic Voting Technology Workshop (EVT 2007)*. August.
- Johansson, P., L. Hall, and S. Sikstrom. 2008. "From Change Blindness to Choice Blindness." *Psychologia* 51(2): 142–155.
- Kahnemann, D. 2011. *Thinking, Fast and Slow*. New York, NY: Farrar, Straus and Giroux.
- Lindeman, M. and P.B. Stark. 2012. "A Gentle Introduction to Risk-Limiting Audits." *IEEE Security and Privacy* 10(5): 42–49.
- National Academies of Sciences, Engineering, and Medicine. 2018. *Securing the Vote: Protecting American Democracy*. Washington, DC: National Academies Press. September.
- Norden, L., M. Chen, D. Kimball, and W. Quesenberry. 2008. "Better Ballots." *Brennan Center for Justice*. July 21. <<http://www.brennancenter.org/publication/better-ballots>>.
- Office of the Minnesota Secretary of State. 2009. "Minnesota's Historic 2008 Election." In *Minnesota Legislative Manual (Blue Book)*, 2009–2010 ed. Saint Paul, MN: Office of the Minnesota Secretary of State. <<https://www.sos.state.mn.us/media/3078/minnesotas-historic-2008-election.pdf>>.
- Perez, E. 2019. "Georgia State Election Technology Acquisition: A Reality Check." *OSET Institute Briefing*. March. <https://trustthevote.org/wp-content/uploads/2019/03/06Mar19-OSETBriefing_GeorgiaSystemsCostAnalysis.pdf>.
- Rayner, K. 2009. "Eye Movements and Attention in Reading, Scene Perception, and Visual Search." *Quarterly Journal of Experimental Psychology* 62(8): 1457–1506.
- Reason, J. 2009. *Human Error* (20th Printing). New York, NY: Cambridge University Press.
- Rivest, R.L., and J.P. Wack. 2006. "On the Notion of 'Software Independence' in Voting Systems." White paper. July. <<https://www.nist.gov/system/files/si-in-voting.pdf>>.
- Rivest, Ronald L. 2008. "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.
- Rivest, Ronald L., and Madars Virza. 2016. "Software Independence Revisited." In *Real-World Electronic Voting*, eds. Feng Hao and Peter Y.A. Ryan, 19–34. Boca Raton, FL: Auerbach Publications.
- Ryan, P.Y.A., D. Bismark, J. Heather, S. Schneider, and Z. Xia. 2009. "Prêt à Voter: A Voter- Verifiable Voting System." *IEEE Transactions on Information Forensics and Security* 4(4): 662–673.
- Stark, P.B. 2008. "Conservative Statistical Post-Election Audits." *Annals of Applied Statistics* 2(2): 550–581.
- Stark, P.B. 2009. "Risk-Limiting Postelection Audits: Conservative P-values from Common Probability Inequalities." *IEEE Transactions on Information Forensics and Security* 4(4): 1005–1014.

- Stark, P.B. 2018. "An Introduction to Risk-Limiting Audits and Evidence-Based Elections." Testimony prepared for the California Little Hoover Commission. July 2. <<https://www.stat.berkeley.edu/~stark/Preprints/lhc18.pdf>>.
- Stark, P.B. 2019. "There Is No Reliable Way to Detect Hacked Ballot-Marking Devices." *ArXiv.org*. August. <<https://arxiv.org/abs/1908.08144>>.
- Stark, P.B., and D.A. Wagner. 2012. "Evidence-Based Elections." *IEEE Security and Privacy* 10(5): 33–41.
- U. S. Election Assistance Commission. 2007. *Effective Designs for the Administration of Federal Elections*. June. <https://www.eac.gov/sites/default/files/event_document/files/EAC_Effective_Election_Design.pdf>.
- U.S. Election Assistance Commission. 2017. *Voluntary Voting Systems Guidelines 2.0*. September. <https://www.eac.gov/sites/default/files/TestingCertification/2020_02_29_vvsg_2_draft_requirements.pdf>.
- Verified Voting Foundation. 2018. "The Verifier—Polling Place Equipment—November 2018." *Verified Voting*. November. <<https://www.verifiedvoting.org/verifier/>>.
- Verified Voting Foundation. 2020. "The Verifier—Polling Place Equipment—November 2020." *Verified Voting*. February 8 (retrieved). <<https://www.verifiedvoting.org/verifier/>>.
- Wallach, Dan S. 2019. "On the Security of Ballot Marking Devices." *ArXiv.org*. December. <<https://arxiv.org/abs/1908.01897>>.
- Wixted, J.T., and G.L. Wells. 2017. "The Relationship Between Eyewitness Confidence and Identification Accuracy: A New Synthesis." *Psychological Science in the Public Interest* 18(1): 10–65.

Address correspondence to:

Andrew W. Appel
 Department of Computer Science
 Princeton University
 35 Olden Street
 Princeton, NJ 08540
 USA

E-mail: appel@princeton.edu

Received for publication December 27, 2019; received in revised form February 14, 2019; accepted April 4, 2020; published online June 17, 2020.