Received 1/10/2020 4:37:23 PM Commonwealth Court of Pennsylvania

Filed 1/10/2020 4:37:00 PM Commonwealth Court of Pennsylvania
674 MD 2019

John F. Murphy (PA Bar No. 206307)
Lesley M. Grossberg (PA Bar No. 208608)
Jeanne-Michele Mariani (PA Bar No. 327000)
BAKER & HOSTETLER LLP
2929 Arch Street
Cira Centre, 12th Floor
Philadelphia, PA 19104-2891
Tel: (215) 568-3100

Ronald Fein (Pro Hac Vice pending)
John Bonifaz (not seeking Pro Hac Vice admission)
Ben Clements (not seeking Pro Hac Vice admission)
FREE SPEECH FOR PEOPLE
1320 Centre St. #405
Newton, MA 02459

Counsel for Petitioners

NATIONAL ELECTION DEFENSE COALITION, CITIZENS FOR BETTER ELECTIONS, RICH GARELLA, RACHEL A. MURPHY, CAROLINE LEOPOLD, STEPHEN STRAHS, KATHLEEN BLANFORD, SHARON STRAUSS, ANNE C. HANNA, RAPHAEL Y. RUBIN, ROBERT F. WERNER, SANDRA O'BRIEN-WERNER, THOMAS P. BRUNO, JR., ROGER DREISBACH-WILLIAMS, and JEFF R. FAUBERT,

Petitioners,

v.

KATHY BOOCKVAR, SECRETARY OF THE COMMONWEALTH,

Respondent.

COMMONWEALTH COURT OF PENNSYLVANIA

ORIGINAL JURISDICTION

Docket No.: 674 MD 2019

NOTICE TO PLEAD

You are hereby notified to file a written response to the enclosed Application For Special Relief In The Nature Of A Preliminary Injunction within twenty (20) days, or within the time set by order of the court, of service hereof or a Judgment may be entered against you.

Respectfully submitted,

BAKER & HOSTETLER LLP

Dated: January 10, 2020

/s/Lesley Grossberg
John Murphy
Lesley Grossberg
Jeanne-Michele Mariani
2929 Arch Street
Cira Centre, 12th Floor
Philadelphia, PA 19104-2891
T: (215) 568-3100
F: (215) 568-3439
johnmurphy@bakerlaw.com
lgrossberg@bakerlaw.com
jmariani@bakerlaw.com

Counsel for Petitioners

John F. Murphy (PA Bar No. 206307)
Lesley M. Grossberg (PA Bar No. 208608)
Jeanne-Michele Mariani (PA Bar No. 327000)
BAKER & HOSTETLER LLP
2929 Arch Street
Cira Centre, 12th Floor
Philadelphia, PA 19104-2891
Tel: (215) 568-3100

Ronald Fein (Pro Hac Vice pending)
John Bonifaz (not seeking Pro Hac Vice admission)
Ben Clements (not seeking Pro Hac Vice admission)
FREE SPEECH FOR PEOPLE
1320 Centre St. #405
Newton, MA 02459

Counsel for Petitioners

NATIONAL ELECTION DEFENSE COALITION, CITIZENS FOR BETTER ELECTIONS, RICH GARELLA, RACHEL A. MURPHY, CAROLINE LEOPOLD, STEPHEN STRAHS, KATHLEEN BLANFORD, SHARON STRAUSS, ANNE C. HANNA, RAPHAEL Y. RUBIN, ROBERT F. WERNER, SANDRA O'BRIEN-WERNER, THOMAS P. BRUNO, JR., ROGER DREISBACH-WILLIAMS, and JEFF R. FAUBERT,

Petitioners,

v.

KATHY BOOCKVAR, SECRETARY OF THE COMMONWEALTH,

Respondent.

COMMONWEALTH COURT OF PENNSYLVANIA

ORIGINAL JURISDICTION
Docket No.: 674 MD 2019

PETITIONERS' APPLICATION FOR SPECIAL RELIEF IN THE FORM OF A PRELIMINARY INJUNCTION UNDER PA. R.A.P. 1532

Petitioners, by and through their counsel, hereby move pursuant to Rule 1532(a) of the Pennsylvania Rules of Civil Procedure for special relief in the form of a preliminary injunction requesting that the Commonwealth be: (1) enjoined from using the ExpressVote XL in any election; (2) required to decertify the ExpressVote XL; and (3) ordered to implement replacement systems that are not in violation of the Pennsylvania Election Code or the Pennsylvania Constitution in order to maintain the integrity of Pennsylvania's electoral system and its democracy as a whole. In support of their application, Petitioners hereby incorporate the Verified Petition for Review filed in this action on December 12, 2019. Petitioners further state the following:

BACKGROUND

1. As set forth more fully in the Petition filed on December 12, 2019, as well as the brief in support of this request for special relief in the form of a preliminary injunction, Petitioners allege that the certification and use of the ExpressVote XL violates the Pennsylvania Constitution as well as the Pennsylvania Election Code.

2. The ExpressVote XL is a polling place voting device. It is one of several voting machines which were introduced in the last few years which are commonly referred to as all-in-one hybrid voting machines. They are called "all-in-one" because they combine two tasks which are more often performed by two

separate devices: marking a voter's choices on a piece of paper, and tabulating votes from a piece of paper. In an all-in-one hybrid, these two voting processes are contained in a single device.

3. A voter uses the ExpressVote XL by inserting into the device a 4.25-inch wide blank card made of thermal paper. The voter uses the device's touch-operated screen and/or assistive technology to select choices in one or more contests in the current election.

4. Once the voter finishes selecting his choices, he selects the "Print" button and at that point the ExpressVote XL tabulates those choices by creating two versions of the voter's choices on the ballot card: an unreadable bar code version which will be read by the machine to tally the votes, and readable human text below which purportedly represents the same information contained in the bar code.

5. The ExpressVote XL is designed so that the ballot card passes under the print head again and after it has already been inspected by the voter while on the way to ballot box. (Exhibit A, Appel Decl. ¶ 42.) At this point, hacked software can be programmed to record different votes. (Appel Decl. ¶ 43.) This is a severe security flaw: the ExpressVote XL's hardware is designed so that, if it malfunctions or if rogue software is installed, it can alter or print additional votes on the ballot, after the voter approves the ballot for deposit into the ballot box.

Even those voters who inspect their ballot and notice nothing amiss cannot ensure their vote is correctly marked. And election officials auditing or recounting paper ballots cannot be sure they are seeing the same votes that the voter saw. (Id. at ¶ 44.) Put simply, there is no way to ensure that a voter's vote is securely cast and vote totals reflect the will of the electorate.

6. All of the above is in violation of Pennsylvania Election Code, Section 1101-A, 25 P.S. § 3031.1, which was written to ensure that a voter's vote remains secure and that every voting machine provide a permanent physical record of all cast votes. The ExpressVote XL does neither. While the insecurity of the voting machine is its most troubling feature, the machine violates many other sections of the Pennsylvania Election Code and the Pennsylvania Constitution, including Sections 1107-A and 1111-A of the Pennsylvania Election Code, ensuring secrecy in voting and accessibility for those with disabilities.

7. Based on these concerns, in July 2019, before the machines were used in any election, some of the parties to this suit along with other concerned citizens (collectively, the "Petitioners") petitioned the Secretary of the Commonwealth of Pennsylvania ("the Secretary") to reconsider the certification of the machines. (Ex. H, Grossberg Decl. Ex. 6, Reexamination Request Petition ("Petition").) However, the Secretary gave little weight to their concerns and dismissed the petition in a largely perfunctory manner. (Grossberg Decl. Ex. 7, Report Concerning the

Reexamination Results of Election Systems and Software ExpressVote XL, issued by Secretary Boockvar on September 3, 2019 ("Reexamination Report").)

8. Many of the concerns the Petitioners raised came to fruition when the machines were debuted in Philadelphia and Northampton in the November 5, 2019 general election. Several major issues with the ExpressVote XL were reported on and after Election Day. The ExpressVote XL machine incorrectly tabulated votes in numerous contests, and voters reported problems using the touchscreens and difficulty reading the machine-printed ballots to confirm they were correct. (Ex. B, Bruno Decl. ¶ 6; Grossberg Decl., Ex. 9; Grossberg Decl., Ex. 10, In re 2019 Municipal Election, Nov. 5, 2019, at 6:3-23).) Many voters had difficulty verifying their selections. (Ex. D, Hanna Decl.; Ex. E, Morales Decl.) There were security issues with the machines during the November 5, 2019 election, with administrator control panels left open during voting (Ex. G, Rubin Decl.; Hanna Decl.) and poll workers wearing the administrator security code for the machines in plain view on cards around their necks (Ex. C, Garella Decl.). Secrecy in voting was severely compromised, with poll workers needing to enter the voting booth and view a ballot card in order to assist a voter wanting to exercise their right to spoil a ballot (Garella Decl.), and furthermore the post-election commingling of ballot cards that was intended to preserve anonymity of voters was both ineffective and completed in an insecure location with open access to ballot cards (*id.*).

9. Because of their concerns and the above highlighted problems, which will no doubt continue to persist without court intervention, Plaintiffs filed a petition for review with the Commonwealth Court on December 12, 2019 alleging that the insecurities in the ExpressVote XL are in violation of the Pennsylvania Election Code and the Pennsylvania Constitution and deny voters the right to free and fair elections, and the right to suffrage.

INJUNCTIVE RELIEF

10. Pursuant to Pa. R. A.P. 1532(a), this Court may order special relief, including a preliminary or special injunction "in the interest of justice and consistent with the usages and principles of law." The standard for obtaining a preliminary injunction under this rule is the same as that for a grant of a preliminary injunction pursuant to the Pennsylvania Rules of Civil Procedure. *Shenango Valley Osteopathic Hosp. v. Dep't of Health*, 499 Pa. 39, 51, 451 A.2d 434, 441 (Pa. 1982); *Commonwealth ex rel. Pappert v. Coy*, 860 A.2d 1201, 1204 (Pa. Commw. Ct. 2004). Preliminary injunctive relief may be granted at any time following the filing of a Petition for Review. *See* Pa. R. A.P. 1532(a).

11. The factors for the Court to consider before issuing a preliminary injunction are as follows: (1) whether the injunction is necessary to prevent immediate and irreparable harm that cannot be adequately compensated by damages; (2) whether greater injury would result from refusing the injunction than

from granting it; (3) whether the injunction will restore the parties to their status as it existed immediately prior to the alleged wrongful conduct; (4) whether plaintiffs are likely to prevail on the merits; (5) whether the injunction is reasonably suited to abate the offending activity; and (6) whether the injunction will not adversely affect the public interest. *Free Speech LLC v. Philadelphia*, 884 A.2d 966, 970 (Pa. Commw. Ct. 2005); *Kessler v. Broder, et al*., 2004 PA Super 200, 851 A.2d 944, 946 (Pa. Super. Ct. 2004) (*citing Summit Towne Centre, Inc. v. Shoe Show of Rocky Mt., Inc*., 573 Pa. 637, 646, 828 A.2d 995, 1001 (Pa. 2003).)

12. As set forth more fully in Petitioners' brief filed herewith, as well as the Petition of December 12, 2019, Petitioners meet all of the requirements for a preliminary injunction in this case.

13. *First*, an injunction is necessary to prevent immediate and irreparable harm. The certification and continued use of the ExpressVote XL threatens such harm by impermissibly burdening the fundamental right to vote and thereby disenfranchising Petitioners and many other Pennsylvania voters.

14. *Second*, greater injury would result from allowing the ExpressVote XL to be used in upcoming elections than from issuing the requested injunctive relief. The Commonwealth has not identified any credible reason that suggests it would be unduly burdened by decertifying the ExpressVote XL when other certified systems can be used in the upcoming 2020 primary. Moreover, the

Commonwealth has averred that security of the election results, and confidence in electoral outcomes is of the highest importance to the state. If that is true, it should be equally as concerned as Petitioners about the continued use of the ExpressVote XL. By contrast, continued use of the ExpressVote XL would surely disenfranchise actual voters and undermine voter confidence in the electoral system.

15. *Third,* Petitioners are likely to prevail on the merits of the underlying claims in this case. The fundamental right to suffrage, free from burden, cannot be abridged absent the most compelling state interest. Here, the state has no rational basis, much less a compelling interest that would suffice to continue to use the ExpressVote XL, while at the same time risking that voters' choices are not correctly recorded and tabulated, and thereby jeopardizing the accuracy of election outcomes. In any event, the test for a preliminary injunction is not whether Petitioners are "guaranteed to prevail," but instead is whether there is sufficient evidence to show that "substantial legal questions must be resolved to determine the rights of the respective parties." *Ambrogi v. Reber,* 2007 PA Super 278, 932 A.2d 969, 980 (Pa. Super. Ct. 2007). The fact that the ExpressVote XL has already proved to be an unreliable voting system is proof that Petitioners' concerns are well-founded and grounded in a legal basis for relief. Moreover, it is evident that many more voters in addition to the named Petitioners would be disenfranchised or

impermissibly burdened should the ExpressVote XL be used again in forthcoming elections.

16. *Fourth*, the requested injunctive relief is reasonably suited to abate the offending activity as it maintains the status quo. *See City of Philadelphia v. Commonwealth*, 837 A.2d 591, 604 (Pa. Commw. Ct. 2003) (granting preliminary injunctive relief and noting that "the public interest lies in favor of maintaining the status quo" pending determination of the merits in the case). "The status quo to be maintained is the last actual and lawful uncontested status, which preceded the pending controversy." *Corbett v. Snyder*, 977 A.2d 28, 43 (Pa. Commw. Ct. 2009). Here the offending activity that the injunction is designed to remedy is the improper certification of the ExpressVote XL. Simply put, no violation will occur if the Court enjoins the Secretary to decertify the machine. The Commonwealth also has multiple avenues to pursue in order to correct the offending activity, all of which are reasonably suited to correct the ill and put no undue burden on the Commonwealth or the affected counties in implementation of a new system.

17. *Finally*, given that an injunction will do nothing more than preserve the right of suffrage in its current form, it will not adversely affect the public interest. By definition, "[t]he public interest ... favors permitting as many qualified voters to vote as possible." *League of Women Voters of N. Carolina v. North Carolina*, 769 F.3d 224, 247–48 (4th Cir. 2014*); see also Purcell v. Gonzalez*, 549

U.S. 1, 4 (2006) (the public has a "strong interest in exercising the fundamental political right to vote" (citations omitted)). And "upholding constitutional rights serves the public interest." *Newsom v. Albemarle Cnty. Sch. Bd.*, 354 F.3d 249, 261 (4th Cir. 2003). A preliminary injunction is also in the public's best interest because it would enhance the integrity of the electoral processes that are "essential to the functioning of our participatory democracy." *Purcell*, 549 U.S. at 4. The Commonwealth cannot seriously demonstrate injury resulting from the relief that Plaintiffs request. The cost and time it would take for the Commonwealth to move to a voter-verifiable secure system do not outweigh Plaintiffs' significant constitutional interests. On balance of the injuries, the facts overwhelmingly favor granting Plaintiffs' injunction.

WHEREFORE, for all of the foregoing reasons and those alleged in the Petition for Review, Petitioners respectfully request that this Honorable Court grant their Application for Special Relief in the Nature of a Preliminary Injunction and enter an order enjoining Respondent, her agents, servants, and officers, and others from certifying the ExpressVote XL voting machine from use in Pennsylvania, and provide any ancillary relief necessary to effectuate the Court's order.

Respectfully submitted,

BAKER & HOSTETLER LLP

Dated: January 10, 2020

/s/Lesley M. Grossberg
John F. Murphy
Lesley M. Grossberg
Jeanne-Michele Mariani
2929 Arch Street
Cira Centre, 12th Floor
Philadelphia, PA 19104-2891
T: (215) 568-3100
F: (215) 568-3439
johnmurphy@bakerlaw.com
lgrossberg@bakerlaw.com
jmariani@bakerlaw.com

FREE SPEECH FOR PEOPLE

/s/*Ronald Fein*
Ronald Fein
John Bonifaz
Ben Clements
Free Speech For People
1320 Centre St. #405
Newton, MA 02459
617-244-0234
rfein@freespeechforpeople.org

Counsel for Petitioners

CERTIFICATION

I certify that this filing complies with the provisions of the *Public Access Policy of the Unified Judicial System of Pennsylvania: Case Records of the Appellate and Trial Courts* that require filing confidential information and documents differently than non-confidential information and documents.

Date: January 10, 2020

/s/ Lesley M. Grossberg
Lesley M. Grossberg (PA 208608)

CERTIFICATE OF SERVICE

I, Lesley M. Grossberg, certify that on January 10, 2020, I caused a true and correct copy of the foregoing titled Petitioners' Application for Special Relief in the Form of a Preliminary Injunction Under Pa. R.A.P. 1532, together with all supporting materials thereto, to be served via the Court's electronic filing system and U.S. first class mail upon the following:

Secretary of the Commonwealth Kathy Boockvar
302 North Office Building, PA 17120
Harrisburg, PA 17120

and

Pennsylvania Office of Attorney General
Strawberry Square
Harrisburg, PA 17120

/s/ Lesley M. Grossberg
Lesley M. Grossberg

John F. Murphy (PA Bar No. 206307)
Lesley M. Grossberg (PA Bar No. 208608)
Jeanne-Michele Mariani (PA Bar No. 327000)
BAKER & HOSTETLER LLP
2929 Arch Street
Cira Centre, 12th Floor
Philadelphia, PA 19104-2891
Tel: (215) 568-3100

Ronald Fein (Pro Hac Vice pending)
John Bonifaz (not seeking Pro Hac Vice admission)
Ben Clements (not seeking Pro Hac Vice admission)
FREE SPEECH FOR PEOPLE
1320 Centre St. #405
Newton, MA 02459

Counsel for Petitioners

NATIONAL ELECTION DEFENSE COALITION, CITIZENS FOR BETTER ELECTIONS, RICH GARELLA, RACHEL A. MURPHY, CAROLINE LEOPOLD, STEPHEN STRAHS, KATHLEEN BLANFORD, SHARON STRAUSS, ANNE C. HANNA, RAPHAEL Y. RUBIN, ROBERT F. WERNER, SANDRA O'BRIEN-WERNER, THOMAS P. BRUNO, JR., ROGER DREISBACH-WILLIAMS, and JEFF R. FAUBERT,

Petitioners,

v.

KATHY BOOCKVAR, SECRETARY OF THE COMMONWEALTH,

Respondent.

COMMONWEALTH COURT OF PENNSYLVANIA

ORIGINAL JURISDICTION
Docket No.: 674 MD 2019

ORDER GRANTING APPLICATION FOR SPECIAL RELIEF IN THE NATURE OF A PRELIMINARY INJUNCTION

AND NOW, this day of , 2020, upon consideration of Petitioners' Petition for Review and Application for Special Relief in the Nature of a Preliminary Injunction, it is hereby **ORDERED** that said Application is **GRANTED.**

IT IS FURTHER ORDERED that Respondent and her agents, servants, and officers and others are hereby: (1) enjoined from using the ExpressVote XL in any election; (2) required to decertify the ExpressVoteXL; (3) and ordered to implement replacement systems that are not in violation of the Pennsylvania Election Code or the Pennsylvania Constitution.

BY THE COURT:

_____

EXHIBIT A

NATIONAL ELECTION DEFENSE COALITION, CITIZENS FOR BETTER ELECTIONS, RICH GARELLA, RACHEL A. MURPHY, CAROLINE LEOPOLD, STEPHEN STRAHS, KATHLEEN BLANFORD, SHARON STRAUSS, ANNE C. HANNA, RAPHAEL Y. RUBIN, ROBERT F. WERNER, SANDRA O'BRIEN-WERNER, THOMAS P. BRUNO, JR., ROGER DREISBACH-WILLIAMS, and JEFF R. FAUBERT, Petitioners, v. KATHY BOOCKVAR, SECRETARY OF THE COMMONWEALTH, Respondent.	COMMONWEALTH COURT OF PENNSYLVANIA ORIGINAL JURISDICTION Docket No. 674-MD-2019

DECLARATION OF ANDREW W. APPEL

I, Andrew W. Appel, declare as follows:

1. I make this declaration of my own personal knowledge and, if called as a witness, could and would testify competently thereto.
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.

10. 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.

11. 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.[1] Therefore it is implausible to say that any computer—or voting machine—is perfectly secure, and as a practical matter a state or county cannot hope to make its computer systems perfectly secure against sophisticated attackers.

12. 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

[1] 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 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.

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.

13. 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 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 error-prone.

14. Most U.S. election jurisdictions (states, counties, or other jurisdictions), including many counties in Pennsylvania, use a system of optical-scan vote counting of hand-marked paper ballots. This is the most secure system that I know of: although the

optical scanner is a computer, and could be hacked to make it cheat, the paper ballots marked by the voters can be recounted by human inspection, yielding the true election outcome (the one indicated by a plurality of voters) no matter what computers may have been hacked.

15. 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.

16. Some election jurisdictions (including many in Pennsylvania, New Jersey, Georgia, Louisiana, and other states) have used Direct Recording Electronic (DRE) voting computers. This is an extremely insecure system: if the voting computers are hacked to misrecord votes and change an election outcome, there would be no visible evidence—and depending on the technology, in some cases no evidence at all. RLAs cannot detect or correct such hacking. 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.

17. Because DRE voting machines are hackable and not recountable, many states, including Pennsylvania, are abandoning the use of DRE voting machines.

18. Some voters cannot mark a paper ballot by hand, because of a visual impairment or motor disability. Federal law since 2002 requires every polling place to have an accessible voting system. In polling places that use hand-marked optical-scan ballots, a typical accommodation used is a Ballot-Marking Device (BMD): this is a computer with a touch-screen and with alternate input methods (such as an audio interface for blind voters or a sip-and-puff interface for voters with severe motor disabilities) that allow voters to indicate their votes; the BMD then prints a ballot that may be counted by an optical scanner.

19. Recently, some election jurisdictions have proposed to use, or have begun using, BMDs for all voters. That is, all voters at the polling place use a touch-screen to indicate their votes, and the BMD prints out a paper ballot that can be counted by an optical scanner.

20. In this declaration I shall explain the severe insecurity of BMDs that *cannot* be corrected by any kind of recount or random audit.

21. Furthermore, some jurisdictions have proposed to use, or have begun using, "hybrid" or "all-in-one" voting machines that combine a BMD with an optical scanner in the same paper path. The ES&S ExpressVote XL is an example of such a machine. In this declaration I shall explain why such machines are *even more insecure* than ordinary BMDs.

22. Like any computer-based voting machine, BMDs can be "hacked," that is, their 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. Logic and accuracy testing (LAT) cannot detect such fraud, because the software can easily be programmed to cheat only on the actual election day.[2]

23. BMDs (and all-in-one machines such as the ExpressVote XL) are insecure because (1) most voters do not inspect the printed-out paper ballot carefully enough to notice whether the BMD 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.

24. This empirical evidence and consequent analysis has been described in a series of scientific papers.

25. DeMillo, Kadel, and Marks[3] observed a real polling place in Tennessee, where voters used touchscreen BMDs 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

[2] 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

[3] 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.

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 ballot, they spent an average of 3.9 seconds inspecting it. There were 18 contests on the ballot, so this is less than ¼ second per contest.

26. Bernhard et al.[4] 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.

27. 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.

28. If most voters don't inspect their BMD-marked ballots, then what are the consequences for the hackability (conversely, auditability) of elections? I analyzed this question, along with Richard DeMillo of Georgia Tech and Philip Stark of the

[4] 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.

University of California, Berkeley.[5] 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 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.

29. 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% corresponding to what the voters indicated on the touchscreen.

30. 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 has been programmed to alter what appears on the screen.

[5] Ballot-Marking Devices (BMDs) Cannot Assure the Will of the Voters, by Andrew W. Appel, Richard A. DeMillo, and Philip B. Stark, April 2019. Available at SSRN: https://ssrn.com/abstract=3375755.

31. 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.

32. You might think, "some sort of audit should catch such hacked BMDS." 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.

33. Therefore, BMD-marked ballots are not meaningfully auditable or recountable: hacked BMDs can cheat in a way that cannot be corrected.

34. 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 auditable and recountable.

35. In the last few paragraphs I have been discussing BMDs that print a paper ballot but do not scan or tabulate. That is, after the voter indicates votes the touch-screen of such a BMD, the machine prints a paper ballot, and the voter carries this paper ballot to a separate machine—an optical scanner—which reads the ballot, tabulates the ballot, and deposits it into a ballot box. Now I will discuss "hybrid" or "all-in-one" BMDs such as the ExpressVote XL.

36. The ES&S ExpressVote XL is a full-face touchscreen voting machine, that (after the voter finishes indicating votes on the touchscreen) prints a paper ballot card and displays it under a plastic screen. I have studied this machine and I have seen on in operation at a voting-system vendors' fair in Trenton, NJ (March 2019). The ballot is printed with human-readable candidate-selections as well as bar codes that can be read by an optical scanner built into the machine. After the voter inspects the ballot and accepts it by pressing a spot on the screen, the ballot is pulled into a ballot box where it is preserved. In that sense the ExpressVote XL is a "hybrid" or "all-in-one" voting machine that combines a Ballot Marking Device (BMD) with an optical scanner and a tabulator.

37. The ExpressVote XL is subject to the same security vulnerability as any BMD: if its computer is hacked to steal some fraction of the votes in a particular contest, and to deliberately mismark the paper ballot, then most voters will not notice. Those voters who do notice will have recourse limited to correcting only their own votes, and therefore the BMD succeeds in stealing the vast majority of votes that it attempts to steal.

38. This is a severe problem, and defeats the purpose of switching from paperless DREs to paper ballots.

39. In addition to this security flaw, the ExpressVote XL has additional security flaws: (1) the voter cannot hold the paper ballot to comfortably read it, (2) the procedure for

voiding mismarked ballots compromises the secret ballot, (3) the printer can print more votes on the ballot after the voter last inspects the paper.

40. Most BMDs print out a paper ballot that the voter can hold in her hand to inspect it. Voters with moderate vision impairments, who may need to use a magnifying glass, a portable scanner, or simply good ambient light, can read such a ballot. In contrast, the ExpressVote XL displays the paper ballot under plexiglass, at an angle the voter cannot control, with lighting that the voter cannot control, in a block-capitals font that is small and difficult to read. This physical arrangement is surely not conducive to voters doing careful inspection of every contest on the ballot, although I am not aware of any scientific study of this particular machine's ergonomics.

41. If and when a voter notices that the paper ballot is incorrectly marked, the voter is supposed to ask a pollworker for assistance. I understand that the design of the ExpressVote XL causes the pollworker, in giving assistance, to enter the booth where the voter's candidate selections are visible. This defeats the secret-ballot protection for this voter. In our hypothetical candidate A-vs-B scenario, the voter might be reluctant to disclose that she was intending to vote for candidate A.

42. The ExpressVote XL has *another* significant security flaw. The ballot-marking printer is in the same paper path as the ballot-box deposit feature. That is, after the voter inspects and accepts the paper ballot, the machine transports the paper ballot *past the print head* on its way to depositing it the ballot box. That means, after the last

time the voter has an opportunity to inspect the paper ballot, the voting machine can print more votes onto the ballot.

43. In its normal operation, with the manufacturer's original software installed, the ExpressVote XL does not print more votes onto the ballot after the voter inspects it and presses "accept." But if a hacker installs fraudulent software in the ExpressVote XL, he can easily program it to illegitimately print such votes. For example, if some voters choose not to vote in the race for Sheriff, the illegitimate software can leave a blank space on the ballot for that contest. The voter inspects the paper ballot and notices nothing amiss. Then when the machine pulls the ballot card up on its way into the ballot box, *past the print head,* the fraudulent software can print a vote for Sheriff in the blank space.[6]

44. This is a severe security flaw: the ExpressVote XL's hardware is designed so that (if rogue software is installed) it can print additional votes on the ballot, *after* the voter approves the ballot for deposit into the ballot box. Election officials auditing or recounting paper ballots cannot be sure they are seeing the same votes that the voter saw.

[6] The manufacturer's software does not print a blank space for an undervote, it prints "NO CANDIDATE SELECTED." But in this scenario, the manufacturer's software is not running, the fraudulent software is running. It is reasonable to assume that the vast majority of voters are not so intimately familiar with the printed paper ballot-card format that they know to expect NO CANDIDATE SELECTED rather than a blank space.

45. I described this class of security flaws in an article[7] published October 22, 2018: "Any voting machine whose physical hardware can print votes onto the ballot after the last time the voter sees the paper, is not a voter verified paper ballot system, and is not acceptable."

46. In summary, paperless DREs are insecure because they are computers that can be hacked, and if hacked can steal votes without the ability of election officials to detect or correct the fraud. It is my understanding that officials of the Commonwealth of Pennsylvania have decided to abandon the use of DREs, principally for this reason.

47. Ballot-marking devices (including the ExpressVote XL) are insecure in much the same way as paperless DREs: they are computers that can be hacked, and if hacked they can steal votes without the ability of election officials to detect or correct the fraud. Some individual voters can detect the fraud, but there is no reliable way they can demonstrate this to election officials in a way that the election official can act upon it. Therefore, the *same* reasons for which an official of the Commonwealth would reject the use of DREs should also apply to BMDs.

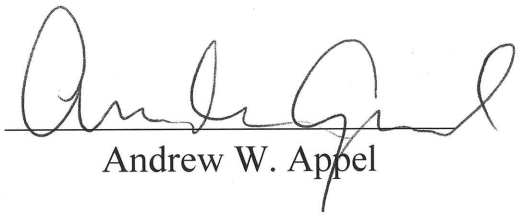
48. All-in-one BMDs (including the ExpressVote XL) are computers that can be hacked, and if hacked they can steal votes without the ability of election officials to detect or correct the fraud. Individual voters can detect the fraud only with great difficulty and sophistication, and (even if they do) there is no reliable way they can

[7] An unverifiability principle for voting machines, by Andrew W. Appel, https://freedom-to-tinker.com/2018/10/22/an-unverifiability-principle-for-voting-machines/, October 22, 2018.

demonstrate this to election officials in a way that the election official can act upon it. Therefore, the *same* reasons for which an official of the Commonwealth would reject the use of DREs should also apply to all-in-one BMDs.

I declare under penalty of perjury under the law of the United States and the Commonwealth of Pennsylvania that the foregoing is true and correct.

Executed on December 27, 2019 in Princeton, NJ.



Andrew W. Appel

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.

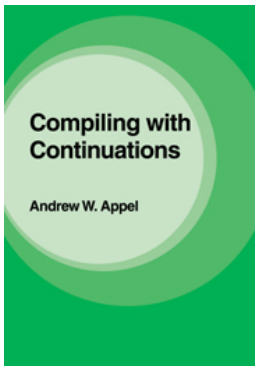
Research Grants

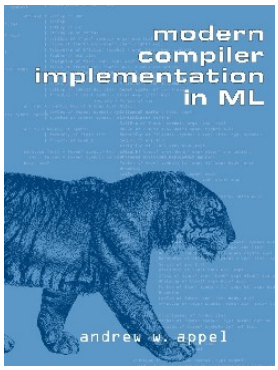
1. *Implementation of an efficient reducer for lambda expressions,* National Science Foundation DCR-8603453, $115,799, 1986-88.
2. Digital Equipment Corporation Faculty Incentive Grant, $180,000, 1986-89.
3. *Unifying compile-time and run-time evaluation,* National Science Foundation CCR-8806121, $123,510, 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.

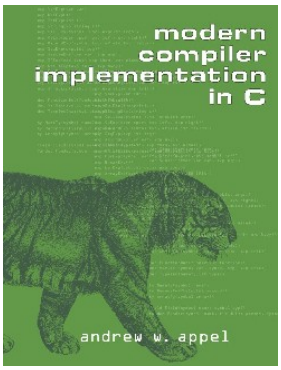
9. *Tools, Interfaces, and Access Control for Secure Programming,* National Science Foundation CCR-9870316, $322,000, 1998-2001 (co-PI).
10. *Scaling Proof-Carrying Code to Production Compilers and Security Policies*, Defense Advanced Research Projects Agency, $3,870,378, 1999-2004.
11. *Applying Compiler 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.

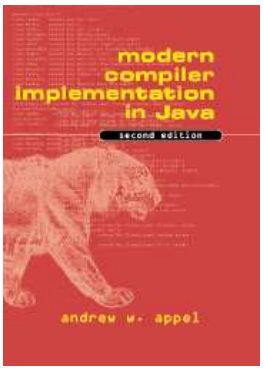
Publications

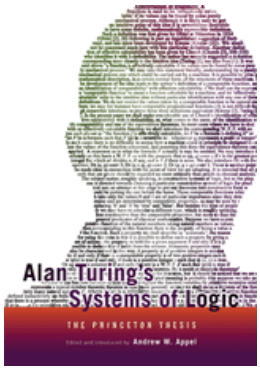
Books, chapters in books

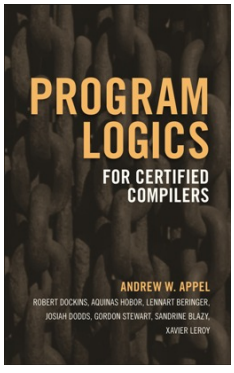


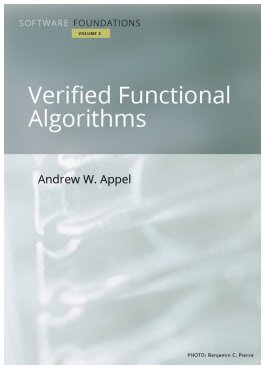












1. ``Garbage Collection," in *Topics in Advanced Language Implementation,* Peter Lee, ed. MIT Press, 1991.
2. *Compiling with Continuations,* Cambridge University Press, 1992.
3. *Modern Compiler Implementation in ML,* Cambridge University Press, 1998.
4. *Modern Compiler Implementation in Java,* Cambridge 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. Hamey. *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 \(em 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. Twenthieth 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.

Workshop and unrefereed conference papers

110. 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.
111. 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.
112. 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.

113. 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.
114. 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.
115. 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

116. 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.
117. SSA is Functional Programming. *ACM SIGPLAN Notices* v. 33, no. 4, pp. 17-20, April 1998.
118. Protection against untrusted code. *IBM Developer Works*, September 1999.
119. 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.
120. Foundational High-level Static Analysis. In *CAV 2008 Workshop on Exploiting Concurrency Efficiently and Correctly*, July 2008.
121. 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.
122. *Freedom-to-Tinker:* 16 articles on the freedom-to-tinker.com blog between 2007 and 2009; 6 articles in 2010; 15 articles in 2011.
123. 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.
124. 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.
125. 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.
126. 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.
127. *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.

Unrefereed papers

128. An Investigation of Galaxy Clustering Using an Asymptotically Fast N-Body Algorithm. Senior Thesis, Princeton University, 1981.
129. Compile-time Evaluation and Code Generation in Semantics-Directed Compilers. Ph.D. Thesis, Carnegie-Mellon University, July 1985.
130. Concise specifications of locally optimal code generators, Princeton Univ. Dept. of Computer Science CS-TR-080-87, 1987.
131. Re-opening closures, Princeton Univ. Dept. of Computer Science CS-TR-079-87, February 1987.

132. Optimizing closure environment representations, by Andrew W. Appel and Trevor Jim. Princeton Univ. Dept. of Computer Science CS-TR-168-88, July 1988.
133. 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.
134. Profiling in the presence of optimization and garbage collection, by Andrew W. Appel, Bruce Duba, and David MacQueen. CS-TR-197-88, November 1988.
135. 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.
136. Emulating Write-Allocate on a No-Write-Allocate Cache, by Andrew W. Appel, CS-TR-459-94, Princeton University, June 20, 1994.
137. Is POPL Mathematics or Science?, by Andrew W. Appel, *ACM SIGPLAN Notices* 27 (4), pp. 87-89, April 1992.
138. Intensional Equality ;=) for Continuations, by Andrew W. Appel, *ACM SIGPLAN Notices* 31 (2), pp. 55-57, February 1996.
139. Ceci n'est pas une urne: On the Internet vote for the *Assemblée des Français de l'Etranger*, by Andrew W. Appel, June 2006.
140. 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.
141. The CompCert Memory Model, Version 2, by Xavier Leroy, Andrew W. Appel, Sandrine Blazy, and Gordon Stewart. INRIA Research Report RR-7987, June 2012.
142. Ballot-Marking Devices (BMDs) Cannot Assure the Will of the Voters, by Andrew W. Appel, Richard A. DeMillo, and Philip B. Stark, April 2019. Available at SSRN: https://ssrn.com/abstract=3375755 or http://dx.doi.org/10.2139/ssrn.3375755

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.

EXHIBIT B

NATIONAL ELECTION DEFENSE COALITION, CITIZENS FOR BETTER ELECTIONS, RICH GARELLA, RACHEL A. MURPHY, CAROLINE LEOPOLD, STEPHEN STRAHS, KATHLEEN BLANFORD, SHARON STRAUSS, ANNE C. HANNA, RAPHAEL Y. RUBIN, ROBERT F. WERNER, SANDRA O'BRIEN-WERNER, THOMAS P. BRUNO, JR., ROGER DREISBACH-WILLIAMS, and JEFF R. FAUBERT, Petitioners, v. KATHY BOOCKVAR, SECRETARY OF THE COMMONWEALTH, Respondent.	COMMONWEALTH COURT OF PENNSYLVANIA ORIGINAL JURISDICTION Docket No. 674-MD-2019

DECLARATION OF TOM BRUNO

I, Tom Bruno, declare as follows:

1. I make this declaration of my own personal knowledge and, if called as a witness, could and would testify competently thereto.
2. I am the Inspector of Elections for Easton's 4th Ward in Northampton County.
3. On the day of the November 5, 2019 election, I was at the polls at First Moravian Church at 225 North 10th Street in Easton from 6:00 AM (set up) to approximately 9:15 PM (pack up).

4. My wife, my youngest daughter, and I all voted on the ExpressVote XL voting machine on November 5, 2019.
5. I voted a straight Democratic ticket.
6. My experience using the machine was that the machine was ultra-sensitive. The mere presence of my finger hovering over a selection list caused an unintentional selection to occur.
7. I reviewed the paper ballot presented in the ExpressVote XL machine before submitting it. It appeared to be correct and so I cast my vote.
8. At the end of the night, we ran the summary tallies from each of the 4th Ward's two ExpressVote XL machines.
9. Both machines showed zero votes for Abe Kassis, a Democratic candidate for county judge.
10. My youngest daughter was Abe Kassis' campaign manager and voted for him, as did my wife and I.
11. The machine showed over thirty straight-party votes for the Democratic ticket, so it was obvious that the machines, despite displaying the correct information on the ballot verification paper, had somehow altered the votes when tabulating them, giving all votes to the only Republican candidate in a 3-candidate race for 2 judge positions.

12. The fact that the display showed my intended vote, but the tally did not reflect that vote, was only made obvious by the severely skewed results.
13. Zero votes for a judicial candidate was obvious and easy to identify as being incorrect. If the machine had only altered a fewer number of votes, the tally would have appeared to be valid and there would be no cause to hand-check the results.
14. The ExpressVote XL machines use a barcode to tally the results.
15. Barcodes are a digital representation that is not discernable to the human eye.
16. The machines have proven that the barcode or the readers in the machine that are supposed to read them, are simply suspect.
17. I want my voting intentions to be properly conveyed in a way that allows me to know with security that what I voted is what is what is recorded.
18. A barcode obviously did not do that in this election.

I declare under penalty of perjury under the law of the United States and the Commonwealth of Pennsylvania that the foregoing is true and correct.

Executed on December 19, 2019 in Easton, Northampton County.

Tom Bruno