Transparency and Access to Source Code in Electronic Voting

Joseph Lorenzo Hall, `joehall@berkeley.edu`
School of Information, University of California at Berkeley

Abstract

We examine the potential role of source code disclosure and open source code requirements in promoting technical improvements and increasing transparency of voting systems. We describe the "enclosure of transparency" of voting technology that has occurred over the course of United States' electoral history, the implications that source code disclosure has for transparency, the negative effects that enclosing transparency has had at different levels and the regulatory and legislative efforts to increase access to source code. We then look at the benefits and risks of open and disclosed source code regimes for voting systems, efforts to provide open source voting systems, existing open source business models that might translate to the voting systems context, regulatory and market barriers to disclosed or open source code in voting systems and alternatives that might exist outside of public disclosure of source code. We conclude that disclosure of full system source code to qualified individuals will promote technical improvements in voting systems while limiting some of the potential risks associated with full public disclosure.

1 Introduction

Elections, like many aspects of society in the United States, have changed dramatically over the course of history. With the growth of urban areas during the last century, and passage of various federal and state laws that specify increased electoral enfranchisement of citizens, we are placing greater and greater demands upon voting technology and election administration. In the past few decades we have started to use computers and networking to further increase efficiency. The most fundamental act of our democracy — the mechanics of casting and counting ballots on election day — that initially took place in plain sight and was fully comprehensible to the franchise now takes place within machines that foreclose observation and obscure this formerly fully comprehensible act. An electoral system that was once highly transparent — supporting public scrutiny and ease of understanding its functions and policies — has undergone an "enclosure of transparency". That is, much like the enclosure movement in English history where public land was increasingly privatized, the requirements to which we hold our voting technologies have resulted in a gradual "fencing in" of transparency.[1] Voting system software is one of the most opaque aspects of electronic voting as it is large, complex and generally unavailable for inspection. Unsurprisingly, academics, activists, election officials and commentators have called for increased access to, and heightened examination of the source code that powers election systems. Efforts to increase access and scrutiny range from source code escrow requirements,[2] independent code reviews,[3] system performance testing,[4] required disclosure of source code to requirements that systems use open source code.[5]

[1] See §2.

[2] Source code escrow involves depositing the source code for a voting system with a third party and/or an election official and stipulating under what conditions the source code can be released. See the discussion of source code escrow in note 42.

[3] A state election official may reserve the right to ask an independent party to do source code review on top of what is done at the federal certification level.

[4] Performance testing involves testing a system in conditions similar to those used on election day.

[5] A note on terminology: There are three important distinctions to make in this discussion. The difference between *open source development* and *releasing commercially developed code under an open source license* is important as these are two modes that we see clearly in voting systems (see discussion of eVACs in §7.1). Open source software is software that is usually developed by a team of volunteers and released under generous licensing terms that allow users to exercise a number of rights, such as copying, modification and distribution, which traditional software licenses withhold. (The Open Source Initiative (OSI) issues and Open Source certification mark to software licenses that follow their Open Source Definition: `http://www.opensource.org/docs/definition.php`.) In contrast to both open source development and releasing software under an open source license, *disclosed source* code allows a much more limited use of source code,

Efforts to broaden the number of individuals with access to the source code of election technology are part of a larger project of increasing the trustworthiness of electronic election systems. This larger project focuses on both technical improvements that increase security, accuracy, privacy, reliability, usability and reforms — at some level independent of technical improvements — that instill confidence in the voting public by facilitating public oversight, comprehension, access and accountability. As such, calls for source code disclosure to the public or to a set of independent experts should be measured along a number of related but independent axes:

- What role could access to voting system source code play in increasing the transparency of voting systems?
- What are the risks and benefits of open source and disclosed source regimes for security and the market?
- If open source code offers measurable benefits, what regulatory and marketplace barriers exist to the development of open source software in the voting systems environment?
- What business methods from the landscape of open source software may translate to voting systems?
- Are there alternatives to public disclosure of code or open source code requirements that might yield similar benefits in technology performance and increased transparency, but minimize potential risks posed by source code disclosure?

This paper examines the potential role of source code disclosure and open source code requirements in promoting technical improvements and increasing transparency of voting systems. Section 2 defines what we mean by transparency and then elaborates on the concept of the "enclosure of transparency" of voting technology. Section 3 explores what implications source code disclosure might have for values associated with transparency. Section 4 details the negative effects that the enclosure of transparency has had at various levels. Section 5 reviews recent efforts to increase the capacity for public scrutiny of voting systems through disclosed and open source code regulation and legislation. Section 6 examines the benefits and risks of open and disclosed source code regimes in the voting systems context and considers additional issues posed where access rules are driven by regulation rather than the market. Section 7 reviews past and current efforts to provide open source voting systems and contemplates which existing open source business models might translate to the voting systems context. Section 8 then considers regulatory and market barriers to disclosed or open source code voting systems. Section 9 reviews what transparency and trustworthy-promoting alternatives might exist outside of public disclosure of source code.

We conclude that disclosure of full system source code to qualified individuals will promote technical improvements in voting systems while limiting some of the potential risks associated with full public disclosure. Acknowledging that this form of limited source code disclosure does not support general public scrutiny of source code, and therefore does not fully promote the transparency goals that we have articulated, we note that in a public source code disclosure or open source code model most members of the public will be unable to engage in independent analysis of the source code and will need to rely on independent, hopefully trusted and trustworthy, experts. Given the potential risks posed by broad public disclosure of election system source code, we conclude that moving incrementally in this area is both a more realistic goal and the prudent course given that it will yield many benefits, greatly minimizes potential risks and provides an opportunity to fully evaluate the benefits and risks in this setting.

2 The "Enclosure of Transparency" of Voting Technology in the U.S.

Early vote casting in the United States was essentially a show of hands or voice vote in front of an official body. In small or even moderately sized towns, it was possible to keep one's own records and do an independent tally of the vote. Of course, this ultimate level of elections transparency in early America had serious implications for voter privacy, coercion and vote selling.[6]

The extreme example of elections in early America allows us to better define what we mean by "transparency": a fully transparent election system is one that supports *accountability* as well as *public oversight*, *comprehension* and *access* to the entire process. This notion of transparency is the core principle of democratic governance; when voters can easily access and comprehend election processes and have the opportunity to observe election-related actions, they are directly exercising their democratic right to hold the system and its pieces accountable.

usually for evaluation purposes only and without permissions to make further copies, modify works or distribute. For example, see VoteHere's license agreement: `http://www.votehere.net/VoteHere_Source_Code_License_2.htm`.

[6]Keller, A. M., Mertz, D., Hall, J. L., And Urken, A. Privacy issues in an electronic voting machine. In *Privacy and Technologies of Identity: A Cross-Disciplinary Conversation*, Katherine J. Strandburg and Daniela Stan Raicu, Eds. Springer Science+Business Media:New York, 2006.

The private ballot, aimed at eliminating coercion, was one of the first major changes to the U.S. voting process and eventually the Australian ballot[7] took hold throughout the vast majority of U.S. states.[8] This helped lessen problems such as biased ballot design, denying ballots to certain groups of people and simple as well as sophisticated forms of vote-selling and voter coercion. Today, all states save West Virginia[9] provide for "secret" or "private" ballots. The requirements to support public scrutiny in a system with secret ballots include ensuring that each voter casts one ballot, that the container in which ballots are cast is initially empty at the beginning of voting, and that no ballots are introduced into the container after the voting is closed. In a paper ballot system, these are largely chain-of-custody concerns and can be ensured by scrutinizing the process and ensuring that there are two people from different parties with the ballot materials at all times.

Due to increasing complexity in counting and casting votes during the last century, voting technology has become mechanized. A number of factors have contributed to this move towards mechanization. Citizens have moved from rural to dense urban areas, causing the number of ballots in cities to increase remarkably. Ballots have become more complex; they often have federal, state and local contests on a single ballot, they often vary from precinct to precinct and they can vary by political party for primary elections.[10] This makes designing and hand-tallying paper ballots difficult and inefficient. Finally, statutory accessibility requirements under state and federal law stipulate accommodations that must be made for voters who don't read or understand English and for voters with physical and mental disabilities.[11]

This mechanization has had profound consequences. On the positive side, election administration has become more efficient as large quantities of paper no longer need to be produced, counted and stored securely. The counting process itself is quicker and many non-English language speakers and persons with disabilities can be accommodated with a single piece of equipment.

Increased mechanization has disadvantages. Flaws with current voting technologies spur concerns that we have been too quick to embrace the productivity-enhancing features of computerized technology while not recognizing the vulnerabilities to which this new technology exposes our electoral system.[12] A more general concern is that the transparency that was at one time a necessary feature of casting and counting votes has been all but lost. Similar to how common property in England during the fifteenth through nineteenth centuries underwent a series of enclosure movements where a public good — common land — was gradually removed from the public sphere, the notion of transparency in the voting franchise has been progressively removed from the electoral franchise.[13] This "enclosure of transparency" has made the mechanisms of the electoral process opaque to the individual voter or even their trusted representative. When "counting votes" consists of running proprietary software to process vote data, voters can no longer "observe" the canvassing process. Nor can regulators or experts, with whom the public places its trust, easily gain access to and evaluate whether votes are being counted as they were intended to be cast.

3 The Implications of Source Code Availability for Transparency

In §2 we defined electoral transparency to have four primary aspects: access, public oversight, comprehension and accountability. Disclosed and open source software support access to the system by allowing a greater sphere of individuals the ability to scrutinize the detailed workings of a voting system. In the case of publicly available source, this access is to all members of the public. With limited disclosure, access is simply increased to a strategically chosen subset of the public to facilitate effective evaluation.

Access to source code supports independent technical evaluation of voting systems that, in turn, facilitates oversight and accountability of software. With access to source code and design documentation, system evaluators can see and analyze each element that goes into building the binary executable which runs on a voting

[7]The Australian ballot provides for a uniform ballot, free from bias in design and presentation, printed by the government and cast in secret.

[8]Id. note 6 at 2.

[9]West Virginia allows "open voting" whereby a citizen may choose to show their marked ballot to whomever they choose (W.V. CONST. ART. IV, § 4, cl. 2.). Interestingly, West Virginia also makes it a crime to sell or buy votes.

[10]For example, in Cuyahoga County, Ohio — which is not required to provide ballots in non-English languages — there were over 6,000 ballot styles provided to voters in the 2006 primary election. See: Candice Hoke, post to the *Election Law listserv*, available at: `http://majordomo.lls.edu/cgi-bin/lwgate/ELECTION-LAW_GL/archives/election-law_gl.archive.0605/date/article-63.html`

[11]Relevant authorities include the Voting Rights Act of 1965, Public Law 89-10 (VRA), The Americans with Disabilities Act of 1990, Public Law 101-336 (ADA), Voting Accessibility for the Elderly and Handicapped Act, Public Law 98-435 and The Help America Vote Act of 2002 Public Law 107-252 (HAVA).

[12]Kohno, T., Stubblefield, A., Rubin, A. D., And Wallach, D. S. Analysis of an electronic voting system. In *IEEE Symposium on Security and Privacy* (2004), pp. 27.

[13]The "enclosure" metaphor has also been extended by legal scholars to apply to recent efforts to reduce the amount of material in the public domain. Boyle, J. The Second Enclosure Movement and the Construction of the Public Domain, 66 *Law and Contemporary Problems* 33-74, Winter-Spring 2003, available at: `http://ssrn.com/abstract=470983`.

system during the election process. They can recompile the code in different manners to facilitate ease of testing and tracing where data goes during processing.

In addition to manual source code review, there are many bug-finding software applications.[14] These tools are developed to automatically find bugs in software by examining source code files or dynamically while the software is running. Evaluators point these tools at large bodies of source code, such as the Linux codebase, and are making much progress at finding common programming errors and vulnerabilities.[15] If voting system software were available to bug-finding researchers, they could examine and perfect their tools further while increasing the integrity of the software. Of course, bug finding is just one example of security-increasing research applications that source code availability could catalyze.

There are also evaluation techniques outside of source code review. It is not impossible to evaluate binary versions of voting system software using techniques from reverse engineering; however, it makes the task more complex and prone to error. [16] There is a rich literature surrounding testing of computerized systems that incorporate unknown, "black box" components[17] and emerging work that seeks to greatly reduce the trusted base in voting systems.[18]

From a systems perspective, evaluation of code is not enough. Even in analyses outside of the ITA process, critical flaws have been found that only become evident when testing the integrated system.[19] We must also include other techniques such as adversarial penetration testing,[20] parallel monitoring,[21] reliability testing and forms of feedback that we have in other areas of computing such as incident reporting and feedback.[22]

Of course, source code availability does not address comprehension; most voters will not gain any more insight into the operation of a voting system when source code has been made available to them. However, the mere fact that it is available and that they or a trusted representative could examine it will increase the level to which they trust these systems.

4 Enclosing Transparency Has Had Negative Effects

This increasing enclosure of transparency has negative effects on a number of levels. First, the voting public cannot see with their eyes or generally comprehend what is happening during the voting process. They have to trust that the voting system works without flaws and that the election official has implemented the voting system correctly.

In a similar vein, election administrators cannot observe what is happening in the depths of their election machinery. Even in cases where the official has access to the technical details of the system, they do not necessarily have the appropriate expertise and resources required to review the system. To provide the level of scrutiny required for their trust, election officials have historically relied on the federal voting system standards and the associated ITA certification process, coupled with any additional State-level evaluation.

However, the federal process also suffers from lack of transparency. The process by which a voting system is

[14]For a partial list of bug-finding tools, see: List of tools for static code analysis, `http://en.wikipedia.org/w/index.php?title=List_of_tools_for_static_code_analysis&oldid=58643351` (last visited June 14, 2006).

[15]For an example of what can be done with automated source code analysis, see: Ashcraft, K., And Engler, D. Using programmer-written compiler extensions to catch security holes. In *SP '02: Proceedings of the 2002 IEEE Symposium on Security and Privacy* (Washington, DC, USA, 2002), IEEE Computer Society, p. 143.

[16]For an example of work that has used binary analysis techniques to uncover vulnerabilities in executable applications, see: Desclaux Fabrice, Skype uncovered: Security study of Skype, EADS CCR/STI/C, November 2005, available at: `http://www.ossir.org/windows/supports/2005/2005-11-07/EADS-CCR_Fabrice_Skype.pdf`

[17]For early work in this area, see: Beizer, B. Wiley, J. Black Box Testing: Techniques for Functional Testing of Software and Systems, *IEEE Software*, 13:5, 98- (1996), available at: `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=536464`

[18]See text in §9.2 and notes 83-85.

[19]See the discussion of the *Hursti II* findings in Hall, note 31.

[20]Penetration testing (sometimes called "Red team" or "tiger team" attacks) involve a simulated attack on a system where the attack team may know everything ("white box" testing) or very little ("black box" testing) about a system and attempt to compromise it in the same manner as would a malicious actor. These types of exercises are common in the testing and implementation of high-integrity systems. For more on penetration testing rationales and methodologies, see: Open Source Security Testing Methodology Manual, available at: `http://www.isecom.org/osstmm/`.

[21]Parallel monitoring, employed during each election now in the State of California, Washington and soon Maryland, involves randomly quarantining a subset of voting machines on election day and voting on them with fake voters and scripted votes to detect bugs, procedural flaws and evidence of possible malicious activity. For more, see: Douglas W. Jones, *Testing Voting Systems: Parallel testing during an election*, The University of Iowa, Department of Computer Science, available at: `http://www.cs.uiowa.edu/~jones/voting/testing.shtml`.

[22]For example, Carnegie Mellon University's Computer Emergency Response Team (CERT) is a computer security incident tracking and response service, see: `http://www.cert.org/`. In response to a question asked by the author at the NIST Voting Systems Threats workshop, EAC commissioners Davisdson and DeGregorio expressed interest in setting up a similar service and process for computerized voting systems.

state and federally approved to be fit for use in a local jurisdiction is widely believed to be inadequate and dysfunctional and is highly opaque. Existing Federal voting system guidelines are weak and out-of-date.[23] Federally certified voting systems have lost votes when used on election day[24] and critical parts of voting systems have made it through federal certification without being examined.[25] The federal certification process relies on Independent Testing Authority (ITA) laboratories to test voting systems for compliance with the federal voting system standards and guidelines.[26] The ITAs are paid by the vendors and all communications and subsequent output from the ITA testing is considered confidential and protected under non-disclosure agreements (NDA) by the vendors.[27] Vendors have claimed that the disclosure of information by the ITAs would implicate their intellectual property rights and compromise the security of their systems.[28] In part, the vendors object to sharing information from the ITA review process based on their desire to maintain "security through obscurity," a principle from computer science that has long been discredited.[29] Source code review by independent, dedicated evaluation teams improves system security; however, the circumstances of the evaluation and relationship between the parties involved should be carefully considered to maximize the utility of evaluation and minimize any undue influence.[30]

Over the past year, there have been a number of cases where the ITA laboratories failed to catch violations of the federal standards.[31] In the face of these failures at the federal level, State and local election officials have had to increase the scrutiny of their systems. Election officials are reluctant to rely on the vendor or ITA to effectively evaluate these systems. They have started to commission their own investigations of particular voting systems using their own independent experts.[32] These officials want to conduct evaluations that are either out of scope or performed poorly in the ITA process. In many cases, especially with additional security testing, access to the source code for voting systems is essential to perform effective evaluation.

5 Regulation and Legislation Relevant to Source Availability

5.1 State-level Disclosed Source Regulation and Legislation

To increase the level of access that they have to voting system source code for evaluation purposes, election officials and state legislatures have started to require that voting system source be disclosed in some form.[33]

In California, the Secretary of State has taken a series of steps to increase the transparency and robustness of voting systems used in the State. The Secretary of State keeps a copy of the source code and binary executables for voting systems and retains the right to perform a full independent source code review.[34] The Secretary exer-

[23]ACCURATE. Public comment on the 2005 voluntary voting system guidelines, 2005, available at: `http://accurate-voting.org/accurate/docs/2005_vvsg_comment.pdf`

[24]*More Than 4,500 North Carolina Votes Lost Because of Mistake in Voting Machine Capacity*, Associated Press / USA Today, November 5, 2004, available at: `http://tinyurl.com/3nhfw`.

[25]NASED letter, "Voting System Memory Card Issues", March 22, 2006, available under "certification" at: `http://www.nased.org/`.

[26]The set of federal standards that are in effect at the time of writing are the FEC's 2002 Voting System Standards (2002 VSS). The EAC's 2005 Voluntary Voting System Guidelines (2005 VVSG) have been approved by the EAC but will not go into effect until January 2008. See: `http://guidelines.kennesaw.edu/vvsg/intro.asp`.

[27]Kim Zetter, E-Voting Tests Get Failing Grade, Wired News, November 1, 2004, (article notes that ITAs cannot discuss specific systems due to NDAs with vendors) available at: `http://www.wired.com/news/evote/0,65535-2.html`.

[28]ITAA letter to Assemblymember Tom Umberg, "OPPOSE: AB 2097", March 22, 2006, on file with author. Similar sentiments were expressed in written testimony to a California State Senate Committee on Elections hearing in February of 2006; see: `http://tinyurl.com/rsk5e`.

[29]Mercuri, R. T. and Neumann, P. G. Security by obscurity, *Communications of the ACM* **46**:11, 160 (2003) available at: `http://doi.acm.org/10.1145/948383.948413`; One of the best discussions of the notion of "security through obscurity" is available on the Wikipedia page for the term. See: Security through obscurity: `http://en.wikipedia.org/w/index.php?title=Security_through_obscurity&oldid=58172204` (last visited June 14, 2006). Full disclosure: the author is one of the many editors of this Wikipedia page.

[30]Lipner, S. B. Security and source code access: Issues and realities. In *SP 00: Proceedings of the 2000 IEEE Symposium on Security and Privacy* (Washington, DC, USA, 2000), IEEE Computer Society, p. 124.

[31]Joseph Lorenzo Hall, "Background on Recent Diebold Election Systems, Inc. (DESI) Vulnerabilities", National Committee for Voting Integrity Briefing for Congressmembers and Staff (2006), available at: `http://josephhall.org/papers/DESI_vulns_background_briefing-20060607.pdf`.

[32]Security Analysis of the Diebold AccuBasic Interpreter, Voting Systems Technology Assessment Advisory Board (VSTAAB), February 14, 2006, available at: `http://ss.ca.gov/elections/voting_systems/security_analysis_of_the_diebold_accubasic_interpreter.pdf`; Linda H. Lamone, Administrator for the Maryland State Board of Elections, letter to Diebold Election Systems, Inc. CEO, available at: `http://truevotemd.org/images/stories//ll-diebold.pdf`. (discussing official's concern and reserving the right to hire an independent expert of their choice to review source code)

[33]There have also been movements to obviate the need for increased transparency, such as the move to require voter-verified paper records (VVPRs). At the time of writing, there are currently 26 states that have enacted legislation requiring Direct-Recording Electronic voting machines to produce a Voter Verified Paper Record to provide an independent check on the voting system's recording functions. See VerifiedVoting.org's Legislation Tracking page: `http://verifiedvoting.org/article.php?list=type&type=13`.

[34]Other provisions relevant to public scrutiny and expert evaluation include: Vendors must establish a California County User Group

cised this right in Spring of 2006.[35]

In the California legislature, there has been one resolution passed and a bill introduced that concerns disclosed and open source software in voting systems. A legislative resolution, ACR 242, was passed in August of 2004 that tasked the California Secretary of State with producing a report on open source code in voting systems.[36] Recently, California Assemblymember Goldberg has introduced AB 2097.[37] This bill would forbid the Secretary of State from approving any voting system for use in California unless "all details of its operating system and specifications are publicly disclosed." It further prevents voting system vendors from exercising any rights against any voter who evaluates the voting system. The Election Technology Council of the Information Technology Association of America, has come out against the bill, as introduced, for a variety of reasons from competitive concerns to intellectual property issues.[38]

In August of 2005, the North Carolina legislature passed SB223/H238 into law which stipulated that all source code used in voting systems certified in North Carolina would have to undergo a variety of evaluations. The provision stated that "all source code" would be made available for review, even that of third party vendors such as the operating system.[39] It is unclear whether or not this statute will be enforced.[40]

Wisconsin recently passed Assembly Bill 627 which, in its original form, required municipalities to provide to any person "the coding for the software that the municipality uses to operate the system and to tally the votes cast."[41] The bill was subsequently amended to stipulate the escrow of voting system software "necessary to enable review and verification of the accuracy of the automatic tabulating equipment".[42]

The intent of legislators and election officials involved in these efforts is to make information about the operation of voting systems publicly available because they think the public has a right to see it, or they see disclosure of source code as a necessary precursor to adequate testing to meet their election responsibilities; or both.

5.2 Federal Legislation

There are a number of Federal bills relevant to source code access. Three bills in Congress address the use of open source or disclosed source software:[43] H.R. 550 (known as "The Holt Bill"), H.R. 939/S. 450 and H.R. 533 would each mandate the use of either open source or disclosed source software in election systems used for

and hold one annual meeting where the system's users are invited to review the system and give feedback and volume reliability testing of 100 individual voting machines under election-day conditions. See: California Secretary of State, "10 Voting System Certification Requirements", available at: `http://ss.ca.gov/elections/voting_systems/vs_factsheet.pdf`.

[35]Id., California VSTAAB, note 32.

[36]"[T]he Legislature hereby requests the Secretary of State to investigate and evaluate the use of open-source software in all voting machines in California and report his or her findings and recommendations to the Legislature." See ACR 242, as chaptered, available here: `http://www.leginfo.ca.gov/pub/03-04/bill/asm/ab_0201-0250/acr_242_bill_20040831_chaptered.html`, Office of the California Secretary of State, "Open Source Software in Voting Systems", 31 January 2006, available at: `http://ss.ca.gov/elections/open_source_report.pdf`.

[37]See AB 2097, "An act to add Section 19213.5 to the Elections Code, relating to voting systems, and declaring the urgency thereof, to take effect immediately.", available at: `http://leginfo.ca.gov/pub/bill/asm/ab_2051-2100/ab_2097_bill_20060217_introduced.html`.

[38]Id., note 28.

[39]See §163-165.7(c), available as passed by both houses of the NC Legislature here: `http://www.ncga.state.nc.us/Sessions/2005/Bills/Senate/HTML/S223v7.html`.

[40]Diebold Election Systems, Inc. was concerned that, among other things, it didn't have the rights to provide access to the source code of third-party software components of its system. It sued the North Carolina Board of Elections to prevent this regulation from taking effect. The case was dismissed as the Court found that there was no dispute as to the language or interpretation of the statute. See: *Diebold v. North Carolina Board of Elections*, unpublished (NC. Super. November 30, 2005), available at: `http://www.eff.org/Activism/E-voting/diebold_order_dismissal.pdf`.

[41]Wisconsin Assembly Bill 627, as introduced, available at: `http://www.legis.state.wi.us/2005/data/AB-627.pdf`.

[42]Wisconsin Act 92, available at: `http://www.legis.state.wi.us/2005/data/acts/05Act92.pdf`. The author knows of at least eight states with escrow requirements in regulation or statute (CA, CO, IL, MN, NC, UT, WI and WA). Unfortunately, it is unclear how many states actually escrow software; some states use the National Institute of Standards and Technology's (NIST) National Software Reference Library (NSRL) as a form of "escrow". However, the NSRL stores binary versions of software products, not source code. See: `http://www.nsrl.nist.gov/`. The conditions for when escrowed software can be accessed and by whom vary but generally protect proprietary information from public disclosure.

[43]There has been no federal electoral legislation since the passage of HAVA in 2002. At the time of writing, there are at least six bills — excluding companion bills — in the U.S. Congress that would substantially reform the conduct of elections on top of the reforms of HAVA. These six bills are: H.R. 550 (text is available at: `http://thomas.loc.gov/cgi-bin/bdquery/z?d109:h.r.00550:`), H.R. 704/S. 330 (text is available at: `http://thomas.loc.gov/cgi-bin/query/z?c109:H.R.704:` and `http://thomas.loc.gov/cgi-bin/query/z?c109:S.330:` respectively), H.R. 939/S. 450 (text is available at: `http://thomas.loc.gov/cgi-bin/bdquery/z?d109:h.r.00939:` and `http://thomas.loc.gov/cgi-bin/bdquery/z?d109:s.00450:` respectively; note the two versions of these bills contain significant differences), H.R. 533/S. 17 (text is available at: `http://thomas.loc.gov/cgi-bin/bdquery/z?d109:h.r.00533:` and `http://thomas.loc.gov/cgi-bin/query/z?c109:S.17:` respectively; note the two versions of these bills contain significant differences), H.R. 278 (text is available at: `http://thomas.loc.gov/cgi-bin/query/z?c109:H.R.278:`) and H.R. 3910 (text is available at: `http://thomas.loc.gov/cgi-bin/query/z?c109:H.R.3910:`). VerifiedVoting.org maintains a comprehensive list of these bills and their differences here: `http://www.verifiedvoting.org/article.php?list=type&type=13`.

federal contests. These are narrow efforts to increase public scrutiny in that they only include source code for systems used in federal elections and it appears that there is little appetite in Congress for electoral reform on top of HAVA.[44]

Both H.R. 550 and H.R. 939/S. 450 would mandate disclosed source for voting system software used in federal elections.[45] The emphasis in these bills is that the source code used to create software used in voting systems be made available to the public. It is unclear from the language of these bills what "disclosed source" would mean exactly; the term is not defined in either bill. H.R. 533 mandates open source, which includes public disclosure, and specifics that the EAC will set standards for such software.[46]

While these bills are motivated by similar concerns, the choice of disclosed or open code is significant. The disclosed source bills provide that software should be available for inspection. The later bill, which uses the term "open source software", leaves the specifics to the EAC to work out. The lack of definitions for these terms is unfortunate given the wide range of possible meanings and possible interpretations for such technical terms.[47] Specifically, disclosed source allows a very narrow subset of rights when compared with open source software licenses.[48]

[44]Congressman Bob Ney, former chair of the Committee on House Administration — which has federal election law jurisdiction — has expressed the sentiment that possible election reform should wait for past legislative action to run its course. See: Speech by Congressman Bob Ney, given at Cleveland State University, Center for Election Integrity on November 30, 2005, available at: `http://cha.house.gov/MediaPages/PRArticle.aspx?NewsID=1146`. This sentiment appears to be the main cause behind why none of the six bills in Congress have gained much traction. While wise in some respects, this mindset neglects the fact that the time cycles involved in development of computerized voting equipment are much quicker than the timeframes included as deadlines in the statutes.

[45]The relevant language in both bills is: "No voting system shall at any time contain or use any undisclosed software." See: H.R. 550 §247(c)(1) and H.R. 939/S. 450 §101(c). The one-word difference is that H.R. 550 would allow the disclosure to any "person" while H.R. 939/S. 450 only allows disclosure to "citizens".

[46]H.R. 533 §329(a) and §299G.

[47]For an appreciation of the variety in open source licensing regimes, browse the Open Source Initiative's (OSI) "Approved License" list `http://www.opensource.org/licenses` and the Free Software Foundation's web page "Various Licenses and Comments about Them" `http://www.gnu.org/philosophy/license-list.html`. Open source licensing covers many licenses, some of which are incompatible with each other. Licenses span a spectrum of very simple — like the modified BSD license: `http://www.opensource.org/licenses/bsd-license.php` — to the very intricate and complex — like the GNU General Public License: `http://www.gnu.org/copyleft/gpl.html`.

[48]Most open source licenses grant or withhold the exclusive rights, granted to creators under copyright law, of copying, modifying and distributing. For detailed inspection of the source code, inspectors would need at least the rights to copy and make modifications. That is, to properly test and debug a program, inspectors will need all source code necessary to build the binary application in a machine-readable format. They would then need to be able to transfer this code to their own build environment, verify that the source code behaves as it purports to, properly build the application and verify that the executable built behaves appropriately and matches the binaries on the target voting systems in the field. Transferring of code, compilation and modification necessary to test source routines implicates the right of reproduction and the right to prepare derivative works or modifications granted by copyright. The right to distribute the source code is not necessarily essential from this perspective as long as the inspecting parties get full access to the code.

5.3 California's "Open Source" Mandate

There is one case where a regulator has required voting system source code be open source. This appears to be the first case of an "open source" mandate by a State in the U.S. where the top election official in California determined that the only solution to a technical catch-22 would be to require a critical piece of code be disclosed. Under recommendations from technical consultants, the Office of the Secretary of State in California issued regulations in November 2003 stating that all electronic voting system vendors would have to provide the functionality required to produce an Accessible Voter-Verified Paper Audit Trail (AVVPAT).[49] An order of March 2004 stated what requirements had to be met for a paper audit trail to qualify as an AVVPAT[50] where AVVPAT was defined as a contemporaneous paper record of a ballot that allowed disabled and non-English speaking voters to vote privately and independently.[51] The biggest surprise in these regulations was the "open source mandate" it included. This part of the regulation provided:

> "All DREs must include electronic verification, as described in the Task Force's report, in order to assure that the information provided for verification to disabled voters through some form of non-visual method accurately reflects what is recorded by the machine and what is printed on the VVPAT paper record. **Any electronic verification method must have open source code in order to be certified for use in a voting system in California.**[52] (bold emphasis added.)

The regulation required an electronic verification mechanism that allows disabled voters to assess through a non-visual interface whether what is printed on the AVVPAT record is consistent with their intended vote.

[49]Kevin Shelley, Secretary of State of the State of California. *Position Paper and Directives of Secretary of State Kevin Shelley Regarding the Deployment of DRE Voting Systems in California* (Nov. 21, 2003). See: `http://www.ss.ca.gov/elections/ks_dre_papers/ks_ts_response_policy_paper.pdf`.

[50]Kevin Shelley, Secretary of State of the State of California. *Standards For Accessible Voter Verified Paper Audit Trail Systems In Direct Recording Electronic (DRE) Voting Systems* (Jun. 15, 2004). See: `http://www.ss.ca.gov/elections/ks_dre_papers/avvpat_standards_6_15a_04.pdf`.

[51]Id., note 49, at 1, 4.

[52]Id., note 49, at 5.

This requires either interpreting the signals sent to the printer or reading directly from the AVVPAT, not from the computer's memory or the electronic record of the vote. The code that interprets the printing signals or reads the AVVPAT must be "open source" per this regulation so that, in the words of David Jefferson, one of the experts that provided input, they would not have "merely transferred the need to trust software from the proprietary vote capture software to proprietary vote verification software."[53]

The regulation left several core terms undefined and the intent unclear. If we take David Jefferson's statement as reflective of the Secretary's goal, this regulation should have been clarified to support the evaluation of the verification software. The regulatory intent here was to ensure that the disabled voter or an organization representing the disabled voters could obtain and inspect the source code of the verification subsystem. They would want to exhaustively inspect the code to make sure that it was accurately verifying the vote from reading the printout or interpreting the signals sent to the printer to produce the printout. The Secretary's decision to require that the source code of this subsystem be open source is logical; however, a clear definition of "open source" is necessary for vendors to build such a system. For example, they will need strict control of what pieces of their intellectual property is included in this piece of software. The Secretary should have aligned the regulatory intent of the AVVPAT order with licensing requirements to establish some minimal licensing criteria for this "open source" software.[54] Then, with a minimal set of licensing requirements, a few representative open source licenses could be chosen and offered as valid licenses under which to develop verification code. This level of detail was not included.

In January of 2005 this requirement was implicitly revoked by new regulations that omitted it.[55] This could have been an interesting experiment in regulatory push of open source; however it seems instead that it was destined to fail without sufficient attention to the issues raised above.

[53]David Jefferson, Chair of the California Secretary of State's Voting Technology Advisory Board, post to the *Voting-Project mailing list*. See: `http://gnosis.python-hosting.com/voting-project/February.2004/0031.html`.

[54]Minimal licensing criteria would be statements such as "The software source code is distributable to any member of the public."

[55]California Secretary of State Elections Division, *Proposed Changes to Accessible Voter Verified Paper Audit Trail (AVVPAT) Standards*, January 14, 2005, available at: `http://www.ss.ca.gov/elections/voting_systems/012005_1b_s.pdf`.

6 Benefits and Risks of Source Availability

Open and disclosed source software present options for improving the performance and public scrutiny of computerized voting systems as they become even more complex. In this section we try to ascertain potential benefits and risks involved in these two models and use this information to evaluate various policy options. Here, we highlight the risks and benefits of both open source and disclosed source software as used in voting systems, by regulatory or legislative mandate or by vendor choice.

If a vendor chooses to use open source software as the basis for the functioning of their system, the most obvious benefit would be the direct access available to source code; anyone who accepts the terms of the open source license will, at least, have the freedom to examine the code. Many more individuals will be able to examine the code using manual or automated analysis. This is one piece necessary to catalyze comprehensive source code review, a key component of the increased security and reliability of source-available software systems.[56]

Disclosed code provides for enhanced access, but does not necessarily support the robust testing that open source code promotes, due to possible restraints on the making of derivative works — such as compiled or modified code — and other manipulations key to certain forms of testing. Disclosed source code regimes provide vendors more flexibility to protect the intellectual property interests than standard open source licenses, which require at a minimum the abilities to copy, modify, prepare derivative works and distribute source code.

Open source software has interesting implications for competition in the market, as the role of copyright and trade secrecy in limiting competition is removed. Therefore a vendor's competitors would be free to modify their code and compete against them with it. Naturally, intellectual property claims will, in general, cease to be a hurdle in commenting on, evaluating, using and procuring these open source voting systems. This is significant given recent efforts by vendors to use IP claims to frustrate oversight and testing of voting systems.[57] Few, if

[56]See Lipner, note 30; "Fuzz testing" — where software products are bombarded with random input to test reliability — has found that source-available software utilizing open source development techniques is considerably more reliable than closed, proprietary products. See: B. Miller, D. Koski, C. Lee, V. Maganty, R. Murthy, A. Natarajan and J. Steidl. Fuzz revisited: A re-examination of the reliability of unix utilities and services. Technical report, Computer Sciences Department, University of Wisconsin (1995), available at `http://citeseer.ist.psu.edu/miller95fuzz.html`

[57]Here are a few examples: In the Fall of 2004, Diebold sent cease-and-desist letters to a number of students who had published an internal email archive that exposed the fact that Diebold had been using uncertified software on their machines. *OPG, Pavlosky & Smith v. Diebold*, 72 U.S.P.Q.2d 1200 (N.D. Cal. Sept. 30, 2004) available at: `http://www.eff.org/legal/ISP_liability/OPG_v_Diebold/20040930_Diebold_SJ_Order.pdf`. Diebold

any, of these cases would have been an issue with an open source voting system as in each case the user of the system would be able to exercise their rights to copy, modify and distribute the software of the system. With disclosed source, we would not have the clear cut case where intellectual property claims become less of an issue, as such claims would now turn substantially on the substance of the disclosed source license the vendor chose to use; it is likely that a vendor would choose to restrict rights to improve its competitive position.

However, there are risks associated with fielding an open or disclosed source voting system. Since computer scientists have yet to find a method for writing bug-free software, public disclosure of the system source code will inevitably result in disclosing vulnerabilities. Voting systems are not the same as general-purpose computing technology. Voting technology is used highly infrequently, runs specialized software and is difficult to upgrade or change without extensive vendor involvement. In the case of voting systems, disclosing information on known vulnerabilities arguably helps would-be attackers more than system defenders.[58] Those tasked with defending voting systems — usually local election officials and their staff — are poorly positioned to shore-up these systems in the case of a serious source code-level vulnerability. Setting aside the fact that most jurisdictions don't have access to system source code, in most states any changes in the system's software will need to be recertified at the Federal and State level before being reinstalled on voting equipment.[59]

Open or disclosed source code voting systems will need to be accompanied by contingency planning in the face of system flaws. Simple flaws may be innocuous enough to allow for the usual running of the election. For serious flaws, such as if there were any suspicion that the flaw will affect the voter experience or the casting, storage and counting of vote data, there will need to be a mechanism to mitigate serious vulnerabilities close to an election. Among the options here would be a "postpone, then patch" strategy where the election in question would be postponed, a fix for the vulnerability developed, the system quickly recertified at the Federal and State level and then the new system used in the postponed election.[60] Another option, more simple than the last, would be for each jurisdiction to be prepared to run the entire election using paper ballots and hand counting. Naturally, jurisdictions using closed source products likely face these problems — known or unknown — now and will want to consider and plan for contingencies; open and disclosed source code raise the stakes of identified flaws.

6.1 The Case of Mandated Source Disclosure

There are risks and some benefits associated with government-mandated public disclosure using either a disclosed source regime or open source licenses. One such risk is that trade secrecy would be de facto eliminated from the highly competitive, small-margin voting systems market. A trade secret is defined as any secret information used in business that gives one a competitive advantage; trade secrecy protection only applies to information that is kept secret.[61] Vendors have asserted that their software contains trade secrets that would no longer be protectable if their software source were disclosed.[62]

The end of trade secrecy in software source code could mean the end for larger companies, which are more sensitive to the smallness of margins, as it will cause a slip of their market position and competitive edge against other larger vendors. If open source software is required, a body of open source software for election management and tabulation will be created that will lower the barriers to entry into the market and necessarily increase competition. The available software will be one piece that new firms will not need to develop in creating a viable voting

has also sent letters and a "product use advisory" to Florida election officials warning them of intellectual property limitations on the testing of their voting systems in conjunction with other vendors systems. See Id., note 23, at 21. In North Carolina, in response to the new legislation discussed in §4.3.2, Diebold sued the State Board of Elections arguing that it could not provide source code to third-party software for the evaluation demanded by the new statute (see note 40).

[58]Swire develops a model of when disclosing security vulnerabilities will help or hinder system defenders: Swire, P. P. A model for when disclosure helps security: What is different about computer and network security? 2 *Journal on Telecommunications and High Technology Law* 163 (2004).

[59]In the past, vendors have "updated" software on voting systems in the field without requesting recertification. After The California Attorney General settled a lawsuit against Diebold Election Systems, Inc. in the Winter of 2004, in part for fielding voting systems which were running uncertified software, this practice seems to have stopped. See: Press Release, California Office of the Attorney General, "Attorney General Lockyer Announces $2.6 Million Settlement with Diebold in Electronic Voting Lawsuit: Settlement Would Resolve False Claims Allegations, Strengthen Security of Equipment", November 10, 2004, available at: `http://ag.ca.gov/newsalerts/release.php?id=843`.

[60]There are unanswered questions about whether or not Presidential elections can be postponed without amending the Constitution. See: Congressional Research Service, "Postponement and Rescheduling of Elections to Federal Office", October 4, 2004: `http://www.fas.org/sgp/crs/RL32623.pdf`.

[61]"A trade secret may consist of any formula, pattern, device or compilation of information which is used in one's business, and which gives him an opportunity to obtain an advantage over competitors who do not know or use it." Restatement of Torts §757, comment b (1939).

[62]Id. ITAA testimony, note 28, "Similarly, software source code, like many other written works (e.g., customer lists, secret formulas for products, strategic plans for future competition and an almost infinite variety of similar materials) can be protected against unauthorized disclosure under state trade secrets laws and with contractual non-disclosure agreements."

system (see §8 for a discussion of other barriers to entry). Either of these possibilities will make it easier for small firms to enter the market, but also may make the market less appetizing for large vendors.

There could be narrower licensing options under a government mandate. That is, if a governmental entity deems it necessary to mandate disclosure, it would seem that they would also specify the terms of such disclosure. This would prohibit vendors from doing their own calculus of what to allow and disallow in the terms of their software license and would mean that they now had to fit their previous business models into the license agreement mandated for the market in which they seek to operate.

Finally, there is an evolving concept of eminent domain in the field of intellectual property, where the government must compensate an individual for taking property. The government "takings" here apply to situations where a vendor's intellectual property is disclosed without their consent or approval. Should vendors be compensated for the release of intellectual property in the source code that runs their systems? The relevant forms of intellectual property implicated in the source code for voting systems are patents, copyright and trade secrets. Patents and copyrights are not much of an issue as both these forms of intellectual property will still be enforceable upon disclosure and there are statutory limits to damages.[63] Claims under the Freedom of Information Act (FOIA) or its state-level equivalents will usually protect proprietary and confidential information.[64]

That leaves the case of trade secrets released against the vendor's wishes. In *Ruckelshaus v. Monsanto Co.*,[65] the Supreme Court found that the disclosure of trade secrets claimed to be held in confidence by the Environmental Protection Agency (EPA) as part of a pesticide registration program was a 5th amendment "taking" of property.[66] The Court ruled that the "taking" existed when Monsanto had a "reasonable investment-backed expectation" of confidentiality and that this was formed when the EPA allowed vendors to mark certain information as trade secret through their registration program.[67] Further, without a reasonable investment-backed expectation, no taking existed. A key feature of the *Ruckelshaus* notion of "takings" is its retroactive nature; that is, the analysis turns on the expectation of confidentiality that the vendor had when submitting information to the government.

For voting systems, this means that any disclosure should be done carefully. That is, with rules or laws that mandate disclosure, any efforts to extend the effects of such policy to source code submissions made under a previous regime would likely run afoul of the *Ruckelshaus* notion of 5th Amendment "taking" of trade secrets. Voting systems vendors will likely not find it difficult to make a showing of "reasonable investment-backed expectation", as past indications show that vendors have been highly protective of their intellectual property.[68] From this analysis, the best course of action would be a non-retroactive policy in which the government clearly stated its intent to disclose system source code and also stipulated that any trade secrets would have to be removed by the vendor prior to submission.

7 Open Source Voting Systems in the Voting Systems Market

If open source voting systems have real advantages compared to closed and disclosed source voting systems, then they should appear in the market much in the way that open source solutions have gained a substantial market presence in other areas of information technology. In this section, we review past and existing efforts to produce an open source voting system and then examine which types of existing open source business models might translate to the voting systems market.

7.1 Open Source E-Voting Projects

There have been a number of efforts to write open source voting code.[69] Most exist purely in software form, but three systems are used or aim to be used in actual elections: Australia's eVACS, The Open Voting Consortium (OVC) and Open Voting Solutions (OVS).

[63] For patents and copyright, 28 USC 1498 provides that a patent or copyright holder can sue the government for "recovery of his reasonable and entire compensation" but cannot enjoin the work being "used by or for" the government. Disclosure of patented, copyrighted software would not correspond to large financial exposure for voting systems vendors; depending on the terms of distribution (limited or public), the availability of the source code for voting system software would not undermine their ability to sell software products or enforce and license their patents.

[64] State equivalents to FOIA in the form of public records acts typically have broad exemptions for confidential information and trade secrets. Exemption 4 of FOIA allows the government to withhold trade secrets under certain circumstances involving FOIA requests. See: Erisman, M. K. The never ending saga of unit prices: To disclose or not to disclose, that is the question. 2005 *Army Law* 138 (2005).

[65] *Ruckelshaus, Administrator, United States Environmental Protection Agency v. Monsanto Co.* 467 U.S. 986 (1984).

[66] Id., at 1003-1004.

[67] Id., at 1010-1014.

[68] See discussion accompanying note 57.

[69] The first quasi-open source software product to be used in U.S. elections was ChoicePlus by Voting Solutions. This software has been used to administer local-level ranked-ballot elections in Cambridge, MA since 1998 and Burlington, VT. It was planned to be released under the GNU GPL in November of 2003 but one small, proprietary piece of code has prohibited the full release of the software under the GNU GPL. Interview with Steve Willet of Voting Solutions, April 7, 2006, on file with author; Jay Lyman, Successful public election joins Diebold, free software, *NewsForge*, April 4, 2006, available at: `http://trends.newsforge.com/article.pl?sid=06/03/23/2040258&tid=136&tid=132`.

Among international efforts,[70] The Australian Capital Territory Legislative Assembly commissioned an electronic voting system in 2000 to be used in the 2001 assembly election.[71] The winning bid, from an Australian firm called Software Improvements, was chosen on the grounds of superior project and quality management as well as increased transparency, as their solution would be freely licensed under the GNU GPL license. Software Improvements designed eVACS to be used on regular PCs that were used during the rest of the year for other purposes.

Aside from the fact that it was the first officially commissioned open source voting system, there are other interesting aspects of the eVACs system. First, while being a GPL'd product, it was not a product of an open source development model; software engineers employed by Software Improvements conducted all development in a highly controlled contribution environment. In fact, when a bug was discovered in the code by outside researchers and brought to the attention of the vendor firm, they developed their own internal fix instead of accepting the outside researchers' fix.[72] Second, the GPL was abandoned for the latest version of the system due to concerns of inadequate Australian legal footing[73] as well as a desire of the firm to protect their intellectual property.[74] However, ACT Electoral Commissioner Philip Greene has said that any future work will have to support the same level of access as what Software Improvements provided with eVACS.[75] Software Improvements is currently in the process of designing a licensing model that would simultaneously solve their concerns while allowing third-party examination and evaluation of the code.

Two groups, The Open Voting Consortium (OVC) and Open Voting Solutions (OVS) have emerged in the U.S. that aim to design or build voting systems with software source code distributed under an open source license. OVS is very new and seems still in the coordination phase of their work but has as its mission to "develop open public specification based voting systems." The OVC, a loose-knit group of activists, information technology professionals and academics, produced a prototype system in 2003 that consisted of demonstration software that ran on commodity computers running the Linux operating system. The OVC's mission now appears to have shifted toward advocacy for the use of open source code in electronic voting systems and away from the production of an electronic voting system.

Given the interest in electronic voting systems powered by open source software it is notable that no working models have fully matured in the current market. I discuss some of the potential reasons for this in Section 8 below. While the verdict is certainly not in on whether the market will independently yield open source powered voting systems, it might now be appropriate to think about other ways of incentivizing open source development so that groups like the OVC can attract the resources needed to produce marketable products. We discuss some possible ideas for this in Section 9.

7.2 Open Source Business Models and the Voting Systems Market

The larger information technology and services sector has seen a substantial growth in business activity directly or indirectly tied to open source software. Is disclosed and open source software something that would naturally arise in the voting systems market? The voting technology market and regulatory environment are sufficiently distinct that a direct translation of current open source business models is questionable. Here, we cover what business models from other open source business endeavors might be applicable in the voting systems market. In Section 8, we highlight some barriers to entry and ongoing business that such an enterprise might face.

A few ways to make money off of open source software used in the IT sector might apply to the business environment surrounding voting systems. Firms such as 10X Software make money off of integrating IT systems into operating environments. A similar idea could be extended to voting, where a system integrator would incorporate open source voting system software and voting hardware to produce a voting solution for a state or local jurisdiction. Some firms, such as Wild Open

[70] The following nations have either posted or claim to have posted voting system software in publicly-accessible forums or to select organizations: Argentina, Venezuela, Estonia and Kazakhstan. See: "Publicación de Software y Documentación", available (in Spanish) here: `http://www.buenosaires.gov.ar/dgelec/index.php?module=pruebaPiloto&file=publicacion`, See: "Auditorías en Venezuela garantizan la integridad del voto", available (in Spanish) at: `http://www.smartmatic.com/noticias_077_2005-18.htm`, See (in Estonian): `http://www.vvk.ee/elektr/docs/Yldkirjeldus-eng.pdf` and documentation/software at: `http://www.vvk.ee/elektr/dokumendid.htm`, Kazakhstan claims to allow review of the source code used to power their voting systems; it is hard to find. The Kazakh elections website (in Cyrillic): `http://election.kz/`.

[71] Clive Boughton and Carol Boughton, "Credible Election Software — eVACS™", white paper on file with author (2005).

[72] Email interview with Carol Boughton of Software Improvements Pty Ltd. (on file with author).

[73] For example, under §68(1) of Australia's Trade Practices Act of 1974, a disclaimer of warranty is void if it does not follow the particular conventions and wording of the Act. See: Fitzgerald, B., And Bassett, G. Legal issues relating to free and open source software. *Essays in Technology Policy and Law (Queensland University of Technology School of Law) 1* (2003).

[74] Software Improvements stated two concerns with releasing code that they've written under the GPL: first, that they would loose any trade secrecy embodied in the code and second, that another firm could use software that they've developed to compete against them.

[75] Email interview with Philip Greene of the ACT electoral commission. (on file with author).

Source, structure their business around targeted development of open source software. A software firm could be hired by a jurisdiction to add, fix or modify certain features of an open source voting system to their own specifications. This could ensure that specific functionality, such as supporting Instant Runoff Voting, was available in the technology that the customer was going to purchase. This also has the benefit that a feature could be added to the software before the open source voting system as a whole was certified and minimize the costs of having to re-certify a base system with the contracted modifications. Dual licensing is where a company offers the same software under two different software licenses, usually one being free software or open source and the other being a commercial license.[76] This can allow their product to benefit from some aspects of open source development while also allowing their customers, commercial and non-commercial, flexibility in their licensing options. For example, MySQL AB offers its MySQL database software freely under the terms of the GNU GPL, but also allows companies to purchase commercial licenses that permit them to deviate from the terms of the GPL. In the voting systems context, a vendor could offer its software for free under a disclosed or open source license, but then charge commercial users to build variants. Companies could use the open source software simply to sell their hardware. That is, with open source software running their voting hardware, they can devote more resources to ensuring that their voting hardware is innovative and as cutting-edge and economical as their customers demand. For example, to concentrate their efforts at selling their high-quality hardware, Apple computer has embraced open source software as the core of their Mac OS X operating system.[77]

Some ways that companies use to make money off of open source do not translate well to the voting systems market. For example, Google's business strategy involves running optimized web search services on server clusters running the Linux operating system. Given the concerns and problems with networking in election systems, it would be difficult for a company to make money off of running open source voting software remotely. IBM sells proprietary software that works on top of or in concert with open source software. A company that tried to do this in the voting market would have to marshal each version of its software package through certification, and then it would only be partially open as a whole.

8 Barriers to Open Source Voting Systems

In addition to the restricted environment for open source business models discussed in the last section, there are also significant regulatory, economic, organizational and perceptional barriers to the use and development of open source software in the voting systems market. In terms of voting system regulation, any changes in system source code trigger system recertification at all levels. Unlike traditional open source software where the ability to change the software frequently is important, open source voting system software development would have to operate differently and take into account that once a product is out on the market, it will be very difficult to change or "patch" the software. In addition, federal and most state certification processes are evaluations of an end-to-end system; it will be insufficient to simply develop the software, as any successful certification will have to include hardware, documentation, and procedures in addition to the software.

Even with sufficient attention to planning and development, it will still be difficult for small firms or non-profits to get a foothold in the elections systems market. It takes quite a bit of infrastructure and financial backing to be able to develop, certify, market, implement and service voting systems. Federal certification alone can take from two months to a year and cost between $150,000 and $400,000 for a single voting system.[78] Contractual performance bonds — where a vendor puts a certain percentage of the cost of a contract in escrow until the system has performed according to a set of criteria in the contract — can be hundreds of thousands to millions of dollars. Due to the nature of state and federal voting systems standards and guidelines, voting systems must be certified as end-to-end voting systems — including precinct-tabulation, data storage and central tabulation — or a vendor of a subsystem has to team up with a larger firm that has the missing pieces and is willing to sponsor a full system certification.[79]

Of course, other pieces of a voting system business outside of code development need to be in place to field a product. To support the requirements of certifying and marketing an end-to-end system, an open source voting systems vendor will need to have a support organization the likes of which no other open source software applications have had to develop. Some open source businesses such as MySQL AB and SugarCRM do have ex-

[76]Of course, under U.S. copyright law, a copyright holder can license their works under as many licenses as they like.

[77]However, as this article goes to press, there are indications that Apple has closed pieces of its software in a strategy to prevent people from running their software on non-Apple hardware. See: Tom Yager, Apple closes down OS X, *InfoWorld*, May 17, 2006, available at: `http://www.infoworld.com/article/06/05/17/78300_21OPcurve_1.html`.

[78]Coggins, C. Independent testing of voting systems. *Communications of the ACM* **47**:10, 34-38 (2004).

[79]Vogue Election Products & Services, LLC. did just this recently when it teamed up with Election Systems and Software (ES&S) to certify and market the AutoMARK ballot marking device.

tensive marketing and support infrastructures for their paying customers, but no open source business produces a product like an end-to-end voting system with on-site support where software, hardware, documentation and procedures are developed, evaluated, marketed, sold and maintained throughout the lifetime of the product.

Finally, in addition to these regulatory, economic and organizational barriers, there are a number of perceptional barriers related to voting system customers that an open source voting system vendor would have to overcome. First, voting system customers — typically local election officials — might not understand the debate around disclosure and system security. The intuitive view is that disclosing system source code will result in a less-secure system. Vendors will have to take care to explain the arguments against "security through obscurity" and how openly published algorithms, for example in cryptography, have proven more robust to attack. Also, to make a sale, open source vendors will need to be able to demonstrate that the organizational structure they choose will be able to support the system over its lifetime or provide alternatives to such support if the vendor goes out of business.

9 Alternatives to Blanket Disclosure that Increase Transparency

Given what we have described as the "enclosure of transparency", that source code access is a key aspect of voting system evaluation and that there are clear risks to public source code disclosure, we now turn to examining alternatives to blanket disclosure of source code. Such alternatives include limited disclosure, increased public access at the Federal level, incentivized or coordinated disclosure and technological mechanisms that support or obviate access.

9.1 Limited Disclosure

It is clear that source code access is key part of effective evaluation. As in the California case,[80] where a critical interface between the paper record and a non-sighted voter was mandated to be open, there are critical pieces of a computerized voting system where public oversight and comprehensibility of the technology is of great importance. The interfaces between ballot presentation and the storage of vote data as well as the myriad of input and output methods are such critical points where maintaining secrecy results in pushing trust from one part of a voting system to another. In the end, openness is a natural and highly efficient way to break this cycle of pushing trust from one system to another. Other areas of critical importance include vote storage, reading and writing. Limited disclosure of this code could achieve many of the benefits of source disclosure while minimizing risks.

[80] See, *supra*, §5.3.

Limited disclosure can be achieved by restricting the scope of code disclosed and the audience to which it is disclosed. That is, what in the code should be disclosed, critical systems (as argued for above) or all the code? Disclosing all the code has the benefit of ensuring that there is no place for malicious or erroneous code to hide. Allowing the public to view all the source code would have the benefits and risks discussed in §6.

Once the decision as to what code is disclosed has been made, we need to decide who gets to see it. As in the federal open source and disclosed source bills discussed previously, do we allow all the public to acquire the voting systems code that will run our election or do we limit the pool to a select few or a subset of the public? On the contrary, if source code dissemination was controlled by application and contract,[81] the goal of having third-party code review could be achieved without the exposure and intellectual property concerns associated with public dissemination. However, a critical piece of restricted dissemination would be a requirement that all output from such reviews would be publicly available and unredacted to balance the exclusivity of code availability.

9.2 Other Alternatives

A natural approach to increasing voting system transparency would be first to tackle the most obscure aspect of the current system. The Federal testing process (discussed in §4) is the most mysterious and critically obscure step in ensuring voting systems perform according to the federal standards for voting systems. We can infer from increased state-level certification requirements and the fact that numerous vulnerabilities have slipped through federal certification over the past year that the federal evaluation process and the voting system standards do not ensure that a voting system can be used in elections free from serious flaws. A first step in increasing the quality of the federal certification process would be to make the testing plans and full evaluation reports public, perhaps in redacted form.

[81] For example, an individual or organization could have to submit an application attesting to certain competences and sign a legally binding agreement that forbid certain activities. Such pre-requisite competencies could be to have a PhD-level degree in an area such as computer science and experience in system evaluation. Examples of activities to forbid would be to distribute the code further, to compile code flaws that aren't made available to the regulatory agency, to publish non-public reports and to transmit source-level information to a vendor's competitors.

Incentivized disclosure is another option. State governments or a consortium of state governments could decide to hold a contest or post a prize for the first development team to produce a voting system, like the ACT's eVACS, that would be released under a specified open source license. Another interesting model is that of "community source" where a consortium of government entities would agree to donate annual dues and full-time coders to a foundation that would develop, certify, market and support the consortium's voting systems.[82]

Finally, there are technological mechanisms for increasing transparency of voting systems. For example, the move in many states to mandate that DRE voting systems produce a VVPAT is essentially public verification of a record independent of the larger system. This allows the customer to treat the larger voting system as a black box as there will always be a verified indelible record of each vote as cast. In this vein, there is a body of work being developed by researchers that narrows the scope and minimizes the amount of what has to be evaluated. Examples of this work include isolated vote storage systems[83], voting systems with dramatically less trusted code[84], and hardware isolation techniques for security verification[85].

10 Conclusion

There has been an enclosure of transparency surrounding voting technology in the United States with recent efforts to halt the enclosure by increasing access to source code. It is clear that some source code access is needed to support transparency of voting systems. There are risks associated with public disclosure of source code and more substantial risks associated with mandated disclosure. The regulatory, financial, organizational and perceptional barriers to the entry of open source voting system software combine such that the open source business models that are now thriving in other sectors don't easily translate to the voting systems market.

We conclude that disclosure of full system source code to qualified individuals will promote technical improvements in voting systems, while limiting some of the potential risks associated with full public disclosure. Considering the alternatives to blanket disclosure mentioned in §9.2, such as increased access to the Federal process, incentives, collaborative models and technological solutions, we still have not explored all our options. We acknowledge that limited source code disclosure to experts does not support general public scrutiny of source code, and therefore does not fully promote the transparency goals of public oversight, comprehension, accuracy and accountability. However, in a public source code disclosure or open source code model most members of the public will be unable to engage in independent analysis of the source code and will need to rely on independent, hopefully trusted and trustworthy, experts. Given the potential risks posed by broad public disclosure of election system source code, we conclude that moving incrementally in this area is both a more realistic goal and the prudent course given that it will yield many benefits, greatly minimizes potential risks and provides an opportunity to fully evaluate the benefits and risks in this setting

11 Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. CNS-0524745.

A very special thanks to my legal supervisor and mentor, Deirdre K. Mulligan; without her advice and direction, this work would only be a shadow of itself. Discussions with the following people were important in the development of this work: Pam Samuelson, Eddan Katz, David Molnar, Ka-Ping Yee, Pam Smith, Naveen Sastry, Dan Wallach, Michael Shamos and Mitch Kapor.

[82]The SAKAI project uses this "community source" model, where a consortium of higher educational institutions have started to develop their own course management software instead of paying vendors . See: `http://sakaiproject.org/`.

[83]Molnar, D., Kohno, T., Sastry, N., And Wagner, D. Tamper-evident, history-independent, subliminal-free data structures on PROM storage -or - how to store ballots on a voting machine (extended abstract). In *IEEE Symposium on Security and Privacy* (2006).

[84]Yee, K.-P., Wagner, D., Hearst, M., And Bellovin, S. Pre-rendered user interfaces for higher-assurance electronic voting. In *USENIX/ACCURATE Electronic Voting Technology Workshop* (2006).

[85]Sastry, N., Kohno, T., And Wagner, D. Designing voting machines for verification. In *Fifteenth USENIX Security Symposium (USENIX Security 2006)* (August 2006).