An Overview of Electronic Voting and Security

Matt Bishop
Department of Computer Science
University of California, Davis
One Shields Ave.
Davis, CA 95616-8562
bishop@ucdavis.edu

Introduction

There are two types of "electronic voting" or "e-voting." The first type is voting over the Internet, using (for example) home computers. The second type is using electronic voting systems in "standalone" mode, unconnected to any network. This note focuses on the second type. Throughout this note, the term "e-voting" means the second type of electronic voting unless explicitly stated otherwise.

This paper has two themes. The first is to consider what qualities e-voting systems must provide in order to be used effectively. The second is to present information about the state of the art of voting equipment based on information that is publicly available.

The author of this note has been accused of being a computer scientist, never an election official. Thus, this paper provides a discussion of various requirements to illustrate how one would determine desirable qualities. Election officials will have their own perspective, honed by experience, of the importance of these requirements, and can weigh them appropriately. Some seem universal; others may apply only under specific circumstances.

The astute reader will note the order of these themes: requirements and qualities first, then analysis. This is deliberate. Development of all systems proceeds in that fashion. First, one states the requirements to be met; then, one designs and implements the system, showing that the requirements are met at each stem. The third section of this note elaborates on this matter to a greater extent.

This paper is organized as follows. First, we discuss the goal of e-voting systems to derive requirements. We focus specifically on those requirements relevant to e-voting machines used in the context of current elections, and not on the use of e-voting machines to provide support for all of the voting process. The third section discusses how those suggest properties of e-voting systems. The fourth uses available public materials to examine current voting machines in the light of the questions developed in section 3. The last section summarizes the qualities that e-voting systems must preserve to maintain both the integrity of elections and the appearance of integrity of elections.

A word about terminology is appropriate. In an election, there may be many candidates and propositions for a voter to vote on. For example, in the October 7, 2003, California gubernatorial election, there was one recall, 135 candidates for governor, and two propositions. Each voter could cast up to 4 votes (one for or against the recall, one vote for governor, and one "yes" or "no" vote for each of the propositions). In this paper, a "race" is the smallest unit upon which a voter will vote, and a "ballot" is the collection of races to be decided during the election. To continue our example, the California election consisted of four races. Each voter may, or may not,

vote on each race, but if the voter casts multiple votes in a single race, those votes are discarded. We will speak of "voting a race" as meaning that the voter cast a vote in a race, and "voting a ballot" as meaning that the voter cast a vote in one or more races on the ballot. The terminology in this area is unsettled, so this will make clear what we mean.

Properties and Requirements

E-voting machines replace the stylus and card punch mechanisms for voting. Traditionally, in Yolo County, voters were given a ballot consisting of a set of cards and a containing envelope, and the ballot number was recorded as used (but the voter's name is *not* recorded as having used the ballot; the number and the name are kept separate). The voter went to a voting booth. After drawing the curtain, the voter would vote. To vote, a voter took a card and placed it into a machine with a punch on one side. The voter would move the punch over the space indicated for a vote, and apply pressure. The punch would punch out part of the card, indicating the voter's choice. The voter continued, removing cards and inserting new cards (or turning a card over and re-inserting it) until she had cast all votes. She then placed the cards in the envelope so the punches could not be seen. She left the voting booth, and placed the envelope into a ballot box. At the end of the day, election officials tallied the votes.

The goal of using an e-voting machine is to replace the paper ballot and punch mechanism with a virtual, or electronic, ballot. The idea is that the voter would enter her votes using a touch screen or other input mechanism. The e-voting station would count the votes and, at the end of the day, report those counts to the election officials. This speeds the tallying process (as the paper ballots need not be counted) and eliminates the need for paper and punches (or styli). It also eliminates some ambiguity when the punch does not separate the chad from the ballot.

Consider the properties of the first scenario, when paper/punch ballots are used. What qualities of this scheme are important for voting as is done in the United States?

1. The voter must be able to vote. Specifically, there must be one or more usable booths; there must be adequate ballots; and there must be a ballot box to receive the cast votes. The mechanism for voting must be adequately explained, and straightforward to use.
2. The votes are secret. The only one who knows how the voter voted is the voter herself. The election officials never look at the punch cards themselves until the votes are tallied. This is the reason for the enclosing envelope: to hide the punch holes. Further, although one can hand the envelope to an election official (who will place it in the ballot box), most polling stations urge the voters to place it through the slot at the top of the box themselves.
3. The votes are anonymous. There is never an association between a voter and the number of her ballot. The voter's name is checked off when she enters the polling place. She then moves to a second official, who ticks off the next ballot number and tells a third official, who gives the voter a ballot with that number. No record is *ever* kept about which voter was given ballot number 35, for example. This prevents anyone from associating a particular voter with a particular vote.
4. The voter can verify her votes at any point up to her dropping the ballot into the ballot box. In Yolo County, the ballot cards contain the propositions, candidates, and other matter being voted on, so the voter can simply look at the card to determine how she voted. In situations where the cards are more cryptic, with (for example) only numbers, the voter may have to re-

enter the booth and use the voting machine to determine how she voted. But she can do so, up until the point she inserts the ballot into the ballot box.

5. The voter can request a new ballot any time before placing the ballot into the ballot box. Voters make mistakes. When a mistake is made during voting, the voter takes the ballot to an election official and requests another. The mismarked ballot is destroyed without anyone seeing the votes made on it, and the voter is provided a new ballot. The number of the ballot is checked off (so there is a record that the particular ballot was destroyed) and the voter's new ballot number is recorded.
6. The voter can vote a limited number of times per race, and once per ballot. The "limited number of times per race" requirement is enforced by discarding all votes cast in a single race if the voter votes too many times in that race.[1] In Yolo County, the "once per ballot" requirement is enforced by ensuring each voter receives exactly one ballot that can go into the ballot box. Note that this is a *maximum*; a voter can take a ballot, and then decide not to vote by leaving the precinct. But the voter cannot then return and cast her vote.
7. The vote tally is accurate and auditable. Accuracy is basic to any election; if votes are counted incorrectly, the selection of the winner(s) may not reflect the actual choice of the voters. Similarly, should a candidate demand a recount, there must be a mechanism to allow an accurate recount that will correct any errors in the initial tally.

This places seven basic requirements on e-voting systems:

1. The e-voting system must be available. If the systems crash repeatedly, the voters may not be able to use them to vote.
2. The e-voting system must provide a simple to use, easy to understand, and hard to misuse interface for the voter to cast votes.
3. The e-voting machine must not be able to associate votes with a particular voter. This preserves the anonymity of the ballot, as well as the secrecy of any particular voter's vote.
4. The e-voting machine must allow the voter to discard her current vote up to the point that the voter officially casts her vote (equivalent to dropping her ballot into the ballot box).
5. The e-voting machine must prevent a voter from casting more than a limited number of votes per race or one vote per ballot.
6. The voter must be able to verify her vote at any point up to the time it is cast. It is *critical* to note that the check *must* be something that cannot deceive the voter. For example, a second party informing the voter how she voted violates this rule, because the second party could lie (and, of course, it also violates the requirement that the ballot be secret; the second party would know how the voter voted).
7. The e-voting system must accurately tally the votes. The software and hardware must not misrecord votes, even by accident, or inhibit the accurate counting of votes in any way.
8. The e-voting machine must provide an out-of-bands mechanism for verifying the reported vote tally. This meets the auditability requirement of voting, and enables recounts. The reason an out-of-bands mechanism must be used is that, were a recount to consist of having the e-vot-

---

1. Usually, the limit is once per race. However, if six candidates are vying for three seats on a City Council, then "the limit" is three votes for this particular race.

ing systems recount their tallies, and then re-add the tallies of the systems, an e-voting system that misrecorded its votes would not be detected, and the incorrect counts would stand. (It is worth noting that California, which counts punch card ballots using electronic card readers, requires that every county manually recount at least 1% of its precincts, to validate that the electronic card readers are working correctly.)

Given these requirements, certain basic qualities of e-voting systems are apparent. We discuss these in the next section.

Qualities of E-Voting Systems

The e-voting systems used in elections must meet the above requirements. In what follows, we relate these requirements to various information technology qualities that are relevant. We also suggest questions that may be posed to vendors to help determine if their systems meet these requirements.

Quality 1 is availability. E-voting systems must be functioning, with the proper ballot, from the beginning of the election period to the end. Further, as poll workers are trained computer technicians, the e-voting machines must be very simple to configure and run. Given the experience of home computer vendors, a system that can simply be turned on and function correctly seems to be most appropriate. Something that requires repeated or careful configuration is likely either to be down most of the time, or (worse) be configured improperly, leading to an incorrect vote count or an incorrect ballot being present.

This becomes an issue if the e-voting software, which provides an interface for the voter to vote and records the vote, runs on top of a general-purpose operating system. The problem is that a general-purpose operating system is very powerful, and must be configured to support that e-voting software *and only that e-voting software*. The last is very important, because if multiple processes interact on an e-voting system, the possibility for problems during execution are greater than if only the e-voting software runs.[2]

Relevant questions for vendors are:

Question 1. Does your e-voting system run *only* the e-voting software you sell, or does it run the e-voting software on top of an operating system? If the latter, which operating system?

Question 2. How long does your e-voting system stay up once it is running?

Question 3. What is the procedure for getting your e-voting machine ready to use, assuming a proper ballot has already been loaded into it?

Question 1 highlights exactly what is running on the e-voting system. Flaws in the operating system may be exploitable, through avenues other than the Internet, to enable a dedicated and knowledgeable attacker to rig that particular e-voting system. Question 2 speaks to ease of use; if

2. As an example, ATM systems should only run ATM software. One major vendor of ATM systems uses Microsoft Windows CE as the operating system base, and runs the ATM software on that operating system. The ATMs are connected to the bank's computers, which in turn are connected to intranets that ultimately are connected to the Internet. Multiple firewalls and procedures provide protection. Nevertheless, ATMs were infected by Internet worms exploiting features of Windows CE [17]. Had Windows CE not been present, and the ATM software running without an operating system, the infection would not have occurred.

the e-voting system keeps crashing (resetting itself), it may not be usable, forcing voters to use some other means of voting (or denying them their suffrage entirely). Question 3 speaks to ease of set-up. Vendors should not be involved in setting ballots; the procedures should be simple enough for election officials to perform, as should maintenance, installation, field testing and configuration. Otherwise, the vendor's employees will be involved, adding additional people who must be trusted not to make mistakes.

Quality 2 is the dual of one aspect of the first. Just as machines should be simple for election officials to use, so must they be easy enough for the average person to vote on. The average voter knows little to nothing about computers, so the interface must be particularly simple to use. Further, sophisticated voters must not be able to subvert the system by voting in unusual ways or supplying non-authorized tokens where tokens are allowed. This suggests the following questions:

Question 4. How have you tested the human interface on your e-voting system? Could someone who has never seen a computer before use it?

Question 5. How do you handle write-in votes? In particular, is there a limit to the length of the names (or number of names) that applies? What happens if the voter exceeds this limit? What happens if a voter writes in a name that she has voted for?

Question 4 is expressed somewhat strongly, but there are active voters who refuse to use computers, or use them only when absolutely unavoidable (the author's father, a writer, was one such person). E-voting systems must not disenfranchise these voters. The next question speaks to a potential problem, that of limits. A standard technique for attacking systems is to look for limits, then deliberately exceed them and see what happens. This occurs in everyday life, too, as any parent of a teenager can attest. The e-voting systems must be able to handle this type of problem. He other type of limit is where the write-in is used to cast a vote for a candidate listed on the ballot. If the race is to elect 2 out of 3 candidates, and someone votes for "Arnold Schwartzenegger" and also writes in "Arnold Schwartzenegger", the laws of the election must control how, or whether, that vote is counted. The e-voting machine must either handle this case correctly, or provide enough information to the election officials so that they can handle this case. Note that such information must not violate any other requirement, specifically requirement 3, requiring anonymity and secrecy of the other votes.

Property 3 is to disallow the system from recording votes in such a way that the votes can be associated with a particular individual. Note this requires tying the vote to an identity *external* to the system. For example, if the voter number 3428's votes can be associated with voter number 3428, that is acceptable *provided* the number 3428 cannot be associated with (say) Arnold Schwartzenegger or Gray Davis. This is analogous to the paper ballots being numbered. By looking at the cast ballot, election officials can say exactly how voter 3428 voted, but they cannot say, "Arnold Schwartzenegger, *who was voter number 3428,* voted as follows …".

Relevant questions for vendors are:

Question 6. How does your system associate votes with individual voters?

Question 7. Does your authentication/authorization mechanism associate voter identities with the authentication or authorization information?

We will examine Question 6 in another context (revocation of vote) below. The key to question 7 is that voter identities should *never* be associated with authentication or authorization information. Otherwise, there is a way to trace the votes back to the voter's real identity.

Property 4 is that a voter must be allowed to discard her vote up to the point at which it is officially cast. This requires that the e-voting system define the point at which a vote is officially "cast" and cannot be undone. That point must be acceptable to the majority of voters. For example, an e-voting machine that considered a ballot cast once all races were voted on, and that did not provide a mechanism to review the votes in each race once the last race were voted upon, would probably be unacceptable because it makes voting in the last race on the ballot the point of casting. Most voters like to be able to review their ballot after voting in all races but before casting their vote. This is an example of an issue that election officials can best deal with, because of their experience in this area.

This property also requires that the voting system allow the voter to change the vote before that point. This means there must be a mechanism to "undo" votes before they are officially cast. If the software does not allow this, then the e-voting machine will not be able to mimic the use of paper ballots because the voter cannot "tear up" their electronic ballots.

Relevant questions for vendors are:

Question 8. What is the point at which the electronic ballot is officially cast, and the voter cannot redo any portion of the ballot?

Question 9. How do voters change their votes while they are using the e-voting machine?

The answers to both these questions may involve election officials intervening, but in all cases, the intervention must respect the other requirements of voting. If the "Mercuri method" [36] is used, it must be implemented in such a way that the election official cannot see the contents of the ballot in order to nullify it.

Quality 5 requires that at most a limited number of votes per race may be cast. The e-voting machine must enforce this, and it must enforce the rule of one vote per ballot. The former can usually be handled in software, by the e-voting system checking the number of votes in each race. But the issue of ballots requires that any authorization mechanism allow only one ballot per voter. This can be enforced both technically (by deactivating the token once the ballot is cast) and procedurally (by ensuring no voter can use multiple active tokens for authorization).

Relevant questions for vendors are:

Question 10. How do you check that your system enforces the limits on voting in a race?

Question 11. What support (procedures, personnel, *etc*.) must be provided to ensure that no voter can cast multiple ballots?

Question 12. What assumptions does your system make about procedures and support that must be present to prevent multiple ballots from being cast by a single voter.

Question 10 asks about testing procedures. The vendor should supply evidence that this aspect of the e-voting machine has been rigorously tested, including races with write-in candidates, large numbers of candidates[3].

3. Arithmetic on computers can produce surprises. A computer represents numbers as a set of bits. A 3-bit number, for example, may be anything from –4 to 3. But if you add 3 and 3 using 3 bit number sin two's complement arithmetic (a standard type of computer arithmetic), you get –2.

Questions 11 and 12 are similar but distinct. Question 11 asks the vendor to identify the specific support he or she believes is needed to prevent voters from casting multiple ballots. Question 12 delves into assumptions that the programmers, designers, and others who developed the e-voting system made about the environment in which the e-voting machine runs. Ideally, each assumption in the answer to Question 12 should lead to a support mechanism or procedure given in Question 11.

Property 6 requires that the voter be able to verify her vote up to the point at which it is officially cast. Because computers can make mistakes (through maliciousness of the developers, through accident, or through failures), it is imperative that the confirmation come from a medium external to the computer. The use of a voter-verifiable audit trail (VVAT) provides this mechanism, as well as providing an audit capability (see below). Note that the VVAT need not be printed; it can be oral, or use some other means of communication. The critical feature is that it be interpreted independently of the e-voting system, and conform to the rest of the requirements (again, disallowing a second party reading it to the voter).

Relevant questions for vendors are:

Question 13. How can the voter verify that your e-voting system accurately recorded the votes she cast?

Question 14. Does this verification require the intervention of an additional party if the voter cannot read (because of blindness or lack of education)?

Question 13 asks for two things. First, the voter must receive some non-electronic feedback showing her votes, so she can verify them independently of the e-voting system. Secondly, the vendor must provide evidence that the votes on that external medium are in fact the ones recorded. The next quality speaks to this more directly.

Question 14 points out that not all voters can read, and some provision must be made to provide those voters with feedback equivalent to that provided voters who can read. For the blind, perhaps Braille (which is not universal) or, better, something aural would meet this requirement. The latter might also meet this requirement if the voter were illiterate. This is something that election officials have had experience with, and can best assess.

Another important quality is that of accuracy. Votes must be recorded accurately, and tallied accurately, to preserve the rights of voters and keep elections meaningful. Two points must be made here. First, there is no e-voting system that cannot be corrupted. Ignoring the issue of corrupt election personnel (who can rig an election whether or not e-voting machines are used), vendors can insert malicious logic designed to rig an election in such a way that it cannot be detected. In particular, examination of the programs within the machines will show no trace of the compromise, and even if the system is completely rebuilt, the trap will remain.[4] Further, even if the software were (somehow) proved to be correct, if the system were updated using unchecked code, or an installer or operator corrupted the system during installation or routine maintenance, the e-vot-

4. For the technically inclined, one way is to create a corrupt compiler, and recompile it in such a way that the executable contains the trap. The source code is deleted, and the uncorrupted source code is replaced. Now, whenever the compiler is recompiled, or specific programs recompiled, the corrupted compiler will re-insert the trap into the executable. But examination of the source code will not reveal the trap. This was proposed by Paul Karger and Roger Schell in a report in 1976 [28], and popularized by Ken Thompson in his Turing Award lecture [48].

ing system and all its votes could be compromised. In one very important sense, the issue of assurance in software is a *minimal* requirement. Assurance in maintenance and operation is equally critical.

Given that no system can be shown to be error-free, one must balance the risks with the benefits. For example, if the company has a rigorous assurance process, from requirements analysis to delivery, installation, and operation of the system, and can demonstrate its process, the evidence that the system will work correctly is much higher than that provided by a company unwilling to reveal its design documents, implementation details, and that distributes upgrades in a haphazard manner. Therefore, the questions below go to process as well as quality. The next section also explores this issue in more detail.

Question 15. What requirements is your e-voting system designed to meet?

Question 16. How do you know it meets them?

Question 17. Once an e-voting system is in the field, how do you handle updating the software and hardware on that system?

Question 18. What do maintenance personnel do when they work with the e-voting systems?

Question 19. If the system is upgraded or has maintenance work done, how can we verify that it still tallies votes accurately, and meets our other requirements?

Question 15 is the most basic question to the process of procuring e-voting systems. If the systems are not designed to meet the requirements of voting in the precincts and political divisions where those e-voting systems will be used, then they are likely to fail to meet all requirements, and will not work. Question 16 speaks to assurance; what evidence can the vendor provide to support the claims made in the answer to Question 15. Questions 17 through 19 point out that security is a process, not a goal, and that *any* changes to a system considered "secure" requires that the security be revalidated.

Question 18 underlies the critical role of administrators (or maintenance personnel), who have access to the e-voting system. This access gives them the power to alter software, delete it, or install new software. They can also configure e-voting systems, create ballots, obtain intermediate totals, terminate the use of a system in an election, and perform other polling functions. As such, proper procedures in administering (maintaining) systems is just as crucial to the accuracy and security of these systems as is the functionality and assurance of the software.

The final quality is the out-of-bands mechanism for verifying the reported vote tally. This is similar to the requirement that a voter must be able to verify her ballot. The issue here is that, if the e-voting machines make mistakes in counting votes, a recount *using the same data and machines* will give the same erroneous result. For this reason, there must be some way to recount votes that avoids the use of the e-voting systems and the medium on which the votes are stored.

This last observation bears some explaining. Let us say the Crooked Voting Co. sells e-voting systems that record every third Republican vote as Democratic, every fifth Democratic vote as Green, and every tenth Green vote as Libertarian. This corruption occurs when the voter asks that the ballot be cast but before the votes are written to the secondary storage medium (disk or card). After the election, a recount is requested. Recounting the votes from the disks or cards on which the doctored votes were recorded will produce the same, corrupt, tallies. If some external medium, a "post-election audit trail" (created by having voters deposit their verified audit trail entries into a ballot box) were used, then the recount would use the results that individual voters

had approved and that were not corrupted. Hence the requirement for an out-of-bands validation mechanism.

Appropriate questions are:

Question 20. What audit mechanism, or external vote tally record, does your system supply, and how do you know it is correct?

Question 21. If an auditor needs to validate your claims of accuracy, how could he or she use the audit mechanism, or external vote tally record, to verify your claims?

As a sanity check, we now validate our model above against a model that Dr. Peter Neumann presented in 1993 [38]. He listed several "electronic voting criteria". Here, we show how his criteria flow from our requirements. The astute reader will note that Dr. Neumann is speaking *only* about e-voting systems and not the voting process in general, although many of his criteria apply equally well there.

System Integrity. This comes from requirements 7 (accurate vote tally) and 8 (auditability of the count).

Data Integrity and Reliability. This comes from requirements 4 (ability to discard erroneous votes before they are cast), 5 (limited number of votes per race and one ballot per voter), 6 (voter verification of vote), and 7 (accurate vote tally).

Voter Anonymity and Data Confidentiality. This comes from requirement 3 (anonymity and secrecy).

Operator Authentication. This comes from requirements 7 (accurate vote tally) and 8 (auditability of the count). Note it is similar to system integrity, because a key feature of that is the ability to audit activity and trace it to a specific person—in this case, the authenticated, trusted operator.

System Accountability. This derives from requirements 1 (availability), 3 (anonymity and secrecy), 7 (accurate vote tally) and 8 (auditability of the count) because these require rigorous testing and validation to show that the tally is accurate, that the machine functioned correctly, and that anonymity and secrecy were preserved.

System Disclosability. This can be derived from requirement 7 (accurate vote tally).

System Availability. This is simply requirement 1 (availability).

System Reliability. This comes from requirements 4 (ability to discard erroneous votes before they are cast), 5 (limited number of votes per race and one ballot per voter), 6 (voter verification of vote), and 7 (accurate vote tally).

Interface Usability. This is simply requirement 2 (simple user interface).

Documentation and Assurance. This is implicit throughout the requirements, because the vendor must show how the system meets the requirements.

Personnel Integrity. Again, this is implied by requirements 1 (availability), 3 (anonymity and secrecy), 6 (voter verification of vote), 7 (accurate vote tally), and 8 (auditability).

Saltman [44] also notes several criteria, which Neumann summarizes as the following points. Again, we compare our requirements to his.

Conform to Relevant Election Laws. This depends upon the particular laws controlling the voting, and so is not discussed here.

System Must Not Be Shared With Other Applications Running Concurrently. This follows from requirement 1 (availability).

Ballot Images Must Be Retained In Case of Challenges. This is a specialization of requirement 8 (auditability).

Pre- and Post- Election Testing Must Take Place. This follows from requirement 7 (accurate vote tally).

Warning Messages Must Occur During Elections Whenever Appropriate. This follows from requirements 1 (availability), 5 (multiple voting disallowed), 6 (voter verification of vote), and 7 (accurate vote tally).

Would-Be Voters Must Be Properly Authorized. This follows from requirement 1 (availability) and 3 (anonymity and secrecy), the latter because an identifiable voter is improperly authorized.

Handicapped Voters Must Have Equal Access. This follows from requirement 1 (availability) and requirement 6 (voter verification of votes).

It Must Be Possible To Conduct Recounts Manually. This is a specialization of requirement 8 (auditability).

Adequate Training Procedures Must Exist. This follows from 1 (availability), 4 (ability to discard erroneous votes before they are cast), 5 (multiple voting disallowed), 6 (voter verification of vote), 7 (accurate vote tally), and 8 (auditability).

Thus, a quick consistency check confirms that the above requirements are appropriate to apply to e-voting systems.

State of the Art of E-Voting Systems

The following is based upon publicly available information. Access to unpublished documents, or other unavailable information, may change some of these points. Further, the comments are not vendor-specific, except where noted.

Question 1. Does your e-voting system run only the e-voting software you sell, or does it run the e-voting software on top of an operating system? If the latter, which operating system?

The Diebold system GEMS apparently uses the Windows CE operating system ([25], p. 60; [43], p. 333, with the operating system version redacted; [51], p. 103, stating generic Windows; [52], p. 292); further, there are grounds to believe that the operating system is modified from the distributed Microsoft configuration and components (that is, it is not "commercial off-the-shelf" or COTS; see [25], p. 73-74), although at least one state bans such changes after state certification (see [51], p. 111). Fidlar-Doubleday apparently also uses a version of Windows (see [27], p. 164). Apparently, the companies can change the Microsoft Windows operating system *before* certification, but once the voting system is certified, neither the software nor the operating system can be changed. It is unclear whether "change" here refers to only upgrading the operating system, or to reconfigure it as well. But the FEC Standards state that if vendors use COTS operating systems, not "modified for use in the vote counting process" ([27], p. 165) then it need not be

tested.[5] This can lead to problems, especially when features of the operating system conflict with voting requirements. Jones recounts a test in which a voter could determine a previous voter's votes by noticing the current setting of the "push button" displayed on the screen ([27], p. 165). This feature was added to Windows after the e-voting software had been certified. We need to emphasize that, under most conditions, the ability to see what was last "pushed" is a useful feature, so there was no attempt to subvert the electoral process here. But given the particular and peculiar requirements of e-voting systems, the "feature" became a bad bug. Finally, numerous authors have pointed out the ways subversion could occur at the operating system level ([29], p. 212; [27], p. 166; [37], p. 218), so even if the e-voting software were correct, incorrect tallies could be recorded.

Question 2. How long does your e-voting system stay up once it is running?

In Georgia, poll workers cycled power due to a "buffer problem" after three sets of updates were applied to fix errors and crashes ([19], p. 229). In Fairfax County, Virginia, several e-voting machines broke down, were taken to the county government center for repairs, and then returned to service ([8], p. 34; [9], p. 38). Florida has also had problems, notably in Miami where precincts were closed for five hours until e-voting systems obtained from ES&S could be repaired, and where at least 100 would-be voters were turned away ([21], p. 102). During human factors testing, one e-voting system experienced catastrophic failure ([2], pp. 599, 605–606), much to the surprise of the investigators. These reports raise questions about whether all e-voting systems are reliable enough to remain functioning through all of Election Day, and suggest that election officials should have contingency plans in place to handle e-voting system failures.

The Johns Hopkins report ([29], p. 202) suggests that an attacker could forge an administration card and use that to disable an e-voting system. Diebold ([12], p. 246–247) admits the theoretical possibility of such an attack working, but states that "such an attack would be detected instantly" and could be traced back to the attacker by investigating all recent voters (p. 247). Jones ([25], p. 65) points out that the rebuttal assumes an attacker cannot get to a voting machine without signing in, meaning that the defense against this threat is the procedures at the polling station.

Question 3. What is the procedure for getting your e-voting machine ready to use, assuming a proper ballot has already been loaded into it?

E-voting systems rely on procedures to prepare them for use in elections. Diebold has ballots pre-installed and distributed with the AccuVote-TS unit ([13], p. 243) so the ballot must be installed before the machine is sent to the polling station ([43], p. 355). According to a report by the Secretary of State of Washington ([41], p. 181), the GEMS software is given information such as ballot type, candidates, precinct combinations, and so forth. The software then generates the specific programming for each precinct, and downloads it to the PCMCIA card. These are then inserted into the e-voting machine. In Georgia, the election officials insert PCMCIA cards containing the ballots; the systems are then tested and, if they pass the tests, are sealed and delivered to the precincts. On Election Day, election personnel at the precincts verify that the machines

5. The quote is from the 1993 FEC standards. The new FEC standards state that "[u]nmodified, general purpose COTS non-voting software… is not subject to the detailed examinations specified in this section" ([15], Volume II, Section 5.2, p. 5–1). Some states apply more rigorous standards. For example, in Georgia, "any change to either the Microsoft operating system or the Diebold election system voids the State Certification" ([51], p. 111).

have recorded 0 vote totals before they are placed into service ([51], p. 107). However, an interview of someone in Georgia who prepared e-voting machines [39] presents a picture of workers downloading updates from an unsecured FTP site over a telephone line to machines in the warehouses; the article includes a claim that the fixed systems underwent no certification after the upgrade ([39], p. 139). Further, Diebold's training materials for the AccuVote-TS voting system do not include an information security component ([43], p. 321). Harris states that the GEMS User Manual "invites everyone to download files from an unprotected ftp site (page 221 of GEMS User Manual)" ([19], p. 227).

These reports do not provide sufficient detail on the procedures for getting e-voting machines ready to use. For example, they do not discuss how to update a system so that it can be certified, or whether each updated system needs to be recertified, or how to construct the ballot placed onto the PC card. Without these details, the answer to Question 3 cannot be determined. However, the above reports, especially the lack of sound information on information security and the recommendation of dangerous practices, do not lead one to believe the answers to that question will be reassuring.

Question 4. How have you tested the human interface on your e-voting system? Could someone who has never seen a computer before use it?

Testing seems to be haphazard. Jones ([27], p. 180) identifies this as an aspect of current standards that needs to be re-examined constantly. Jones also states that the "computer interfaces on these machines are generally very well designed" ([27], p. 164). Reed states that the audio voting interface was used in a test election at a public hearing ([41], p. 182), but unfortunately, gives no details of the methodology and evaluation. Bederson and Herrnson [2] conducted rigorous testing, and concluded that one particular system (Diebold) has both strengths and weaknesses.

The Diebold system uses smart cards containing information that selects the proper ballot from the ballots programmed into the PCMCIA card in the e-voting system ([41], p. 181). The cards are canceled after each use, and must be reset to be used again ([29], p. 198). There is a dispute about how easily a usable, unauthorized card could be created. Researchers at Johns Hopkins opine that it should not be hard ([29], p. 200; [30], p. 272); Diebold strongly disagrees. Diebold states that the cards are special-purpose cards with unique configurations that could not be obtained without alerting the vendor to a possible attack ([12], pp. 244–245), that any attempt to discover the protocols used by the cards would be detected by poll workers, and that even if the attackers could cast votes in this manner, procedural mechanisms such as voter count reconciliation would detect the problem. Jones ([25], p. 64) points out that poll workers may rely on the smart cards to provide security, and so be less vigilant in checking.

Question 5. How do you handle write-in votes? In particular, is there a limit to the length of the names (or number of names) that applies? What happens if the voter exceeds this limit? What happens if a voter writes in a name that she has voted for?

The Johns Hopkins report states that the write-in candidate's name is stored as an ASCII string ([29], p. 207), but they do not discuss the error handling (if the name is too long for the storage allocated, for example). They do state that "the code reflects an awareness of avoiding such common hazards as buffer overflow" ([29], p. 211), which is a good sign.

If the voter both writes in the name of a candidate appearing on the ballot, and also votes for that candidate on the ballot, e-voting machines do not count the votes. Instead, they report the situation. Jones recounts a test of the Fidlar-Doubleday system in which, as a result of this voting,

the system printed a report with enough information to enable poll workers to apply the relevant law. Unfortunately, the special report contained an appendix with the image of every ballot involved—and by law, this document would be public. The non-write-in votes were encoded; however, it took Jones under a minute to break the code ([27], p. 170). Ideally, the system should only print the write-in vote and the other vote for the candidate. Another question is whether the machines store these votes, and the poll workers need to adjust the recorded totals to account for these irregular votes, or whether the machines do not store the votes and the poll workers must add them. The literature that is publicly available does not discuss this.

Question 6. How does your system associate votes with individual voters?

There should be no association of votes with individual voters. Vendors try to implement this requirement, but sometimes their safeguards can be defeated. The test recounted above would allow association of names with votes if voters were to enter a known name on a write-in line (for example, their own name); this problem is ameliorated if only the irregular votes are printed (and not the whole ballot). The Johns Hopkins report noted that the Diebold program associated a serial number with each vote, and the serial number is unique for each vote ([29], p. 207). The serial number is generated using a linear congruential random number generator (LCG), a standard technique for producing pseudorandom numbers in computer science. Diebold states that the purpose of these numbers is to "pseudo-randomize the order of the ballots" ([12], p. 253). The problem with this argument is that the serial number is recorded with the ballot. If the order in which the voters use the voting machine is known, and the order in which the pseudo-random numbers were generated is known, associating a voter with a vote is trivial. Even if one does not know the order in which the pseudo-random numbers were generated (as one would if, for example, the votes were written to a file in the order they were cast), techniques for attacking LCGs are well known in the literature (see for example [6] and [7]), and reconstructing the order would not be difficult, especially if the increment, multiplier, and modulus were known. Such an attack requires access to both the order in which voters voted at that particular e-voting system, and at least some of the serial numbers. Jones discusses the feasibility of obtaining this information ([25], p. 68), concluding it is possible with an "insider".

The best answer to this question is that vendors are aware of this problem, but one scheme which is used can be defeated, assuming the attacker has access to certain information.

Question 7. Does your authentication/authorization mechanism associate voter identities with the authentication or authorization information?

The answer to this question appears to be that vendors do not do this, with one potential exception (see next paragraph). The Johns Hopkins report states that "no information that can be linked to a voter's identity is included" in the storage of the voter's votes ([29], p. 207), and there is no evidence to the contrary. However, the concern raised for Question 5, above, applies.

The single exception is provisional or challenged ballots. If the polling station is unable to validate a voter's claim to be eligible to vote, the voter will often be allowed to cast a "provisional" ballot. In the paper/punch model, the ballot is places in a sealed envelope (and not the ballot box). The identity of the voter is somehow associated with the envelope) in such a way that opening the envelope does not allow the election official to see the ballot. (In Yolo County, the ballot is placed in an opaque envelope to conceal the votes, and then that envelope is placed into a red envelope on which is written the voter's name and the reason that the vote is provisional. If the vote is counted, the inner envelope is removed and placed in the ballot box, preserving the

anonymity of the voter.) The election officials then either validate the claim (in which case the envelope is opened and the ballot counted) or reject it (in which case the envelope is destroyed). Some jurisdictions use optical ballot scanners to scan in provisional votes, which can then be added to the vote totals (see for example [51], p. 106). Some vendors provide e-voting systems that can handle provisional votes ([11], pp. 264–265; [12], p. 246), but some jurisdictions do not use them (see for example [41], p. 183; the reason is not given). The way in which vendors handle provisional ballots is unknown, as is how they implement the requirement that voter identities cannot be associated with votes.

Hence the answer to this question is unclear, especially for e-voting systems that provide the capability to record provisional ballots.

Question 8. What is the point at which the electronic ballot is officially cast, and the voter cannot redo any portion of the ballot?

Question 9. How do voters change their votes while they are using the e-voting machine?

These questions are closely related. E-voting systems generally allow users to correct any errors in voting before they complete the ballot. Diebold e-voting systems in Georgia, for example, allow the voter to tap on the screen to change a vote, and provide a summary screen before the ballot is counted; the vote can go back to change an vote at the summary screen ([51], p. 114). The Bederson and Herrnson study ([2], p. 609) found that most users found this scheme easy to use, with frequent computer users finding it easier than others. But they noted that if the "Cast Ballot" were pressed by accident, it could not be canceled ([2], p. 603).

Testing performed at Iowa ([27], p. 163–164) showed that repeated voting by a voter for multiple ballots was physically straining for the testers. After casting more than 20 or 30 ballots, the test plan had to be abandoned because the testers made too many mistakes. It is important to emphasize this was a test, and normally voters cast one ballot per election. However, it suggests that if a voter makes repeated errors, correcting them may become tedious and error-prone.

Question 10. How do you check that your system enforces the limits on voting in a race?

The e-voting software keeps the voter from voting more times on each race than is allowed, and prevents the voter from casting multiple ballots. For example, Diebold's e-voting machine disables the smart card used to allow the voter to vote ([43], p. 334). The exact method used to disable the card is, apparently, to change its type ([29], p. 201) by altering a value stored on the card. Without more details of how this is done, if the smart card were programmable and the protocol did not make the smart card's memory directly available to the e-voting system, the program on the card could simply ignore the instruction to disable, and instead change the voter serial number to appear to be a new card. Diebold states that physical controls would make this type of attack ineffective ([12], p. 245); the overvotes would be detected during the process of roster reconciliation, and the bogus voter serial numbers would not show up on the roster. The SAIC report ([42], p. 358) also points out that the e-voting machine makes a loud noise when the card is ejected, implying the election officials would hear the ejection and, if multiple ejections for a single voter occurred, would know the voter is trying to compromise the system. If the polling place is noisy, however, this may not be true.

An interesting question is, "What would happen if the attacker knew the scheme by which the voter serial numbers were generated, and had the card generate valid numbers?" Presumably, these numbers would be duplicated when the legitimate smart cards programmed with those numbers were used. But if the ballot order is scrambled, as the Diebold response to the Hopkins report

implies ([12], p. 253), there would be no way to detect which was the overvote and which was the real vote.

Vendors appear to handle multiple votes in the e-voting machine, but it is not clear their methods work against attacks involving compromised smart cards. Nor is it clear that roster reconciliation will detect *which* votes are overvotes (although it will detect that multiple votes have been detected).

Question 11. What support (procedures, personnel, etc.) must be provided to ensure that no voter can cast multiple ballots?

The response of vendors, and the SAIC review, appears to be that normal procedures such as voter reconciliation and physical security at the polling stations will prevent this problem. However, news organizations have reported that there were insufficient numbers of trained poll workers for some elections (see for example [31], p. 24; [33], p. 11) and raised concerns about future elections (see for example [34], p. 287). Harris raised similar issues, citing confusing instructions from Diebold training manuals ([19], p. 227). Jones has also raised these issues ([25], pp. 64, 65). Indeed, Prof. Williams ([51], p. 115) cites the inability of poll workers "to know how many ballots have been spoiled and ensure that the correct paper ballot is deposited in the ballot box" in his argument against using a voter-verifiable paper trail.

The key issue of this question is, "What happens if the procedures fail?" Can the overvotes be eliminated, or will both valid and overvotes need to be removed? The former answer is acceptable; the latter, not, as it removes legal votes.

Question 12. What assumptions does your system make about procedures and support that must be present to prevent multiple ballots from being cast by a single voter.

The common assumption appears to be a reliance on procedural mechanisms, as noted above. But the reliance on a *single* security mechanism, as is apparently done here, violates a basic principle of security: that access to resources (here, the ballot) requires multiple checks.[6] If the polling station is very busy, it is easy to believe that overworked poll watchers and workers could miss something. This is where a second line of defense should prevent voters from casting multiple ballots, and is the reason that the reassurance of "poll workers will detect problems" is unsatisfying. Using strong encryption, cryptographically strong pseudo-random number generation, and other mechanisms could improve the ability to defeat attackers even if the attacker slips past the observers. So, this question asks the vendors to describe what they base their mechanisms on, something that the current literature does not document well.

Question 13. How can the voter verify that your e-voting system accurately recorded the votes she cast?

A widespread belief is that a voter-verifiable audit trail (such as a paper print-out) provides this assurance (see for example [50], p. 4; [27], p. 152; [52], p. 295; [26], p. 458; [25], p. 98; [29], p. 193–194, 213). The validity of this assumption depends on how votes are actually counted. If the element that the voter actually verifies is counted (such as the paper print-out), then the voter-verifiable audit trail provides the necessary assurance. But if the voter-verifiable is not used for anything other than voter verification that the machine has printed the correct votes, the trail assures nothing.

---

6. This is called the *principle of separation of privilege* in the computer science community [45] and *defense in depth* in other communities.

An example will make this clear. Assume the election is expected to be reasonably close. A bug in the e-voting system causes 10% of the Hackers' party votes to be given to the Developers' party. This swings the election from the Hackers' party to the Developers' party. Unless the Hackers' party calls for a recount (and, in some jurisdictions, can pay for it), *and* that recount uses the voter-verifiable audit trails, the problem will not be discovered. The voter-verifiable audit trail must be used to do the recount, and one party to the election must call for the recount. Unless this occurs at all elections, any bugs in the system will not be noticed and the above assurance will not be present. For this reason, the Mercuri method (in which e-voting machines are used to print paper ballots, and those are counted) has the great advantage of providing the above assurance, and e-voting systems that electronically count votes do not provide the same degree of assurance. As all e-voting systems used today count votes electronically, none provide the same degree of assurance as would an implementation of the Mercuri method.

VoteHere, Inc., announced that it would use third-party software to provide "voters with paperless post-election confirmation of their ballot choices" ([49], p. 435). The problem with such a scheme is that the software merely adds another layer that could be misprogrammed or tampered with. The advantage is that it follows the principle of separation of privilege, assuming the vendor of the validation software is independent of the vendor of the e-voting system. The press release also says that the vendors will be studying the potential addition of an "optional paper-based form of additional voter verification", which would make this method equivalent to having a voter-verifiable paper trail.

The key point is that the voter must be assured that the vote is recorded as made. An e-voting system cannot provide this assurance unless the voter can see the vote recorded in such a way that it cannot be changed before it is recorded, just as a voter can see the votes on a paper ballot in non-e-voting systems, and that those votes cannot be changed before the ballot is placed into the ballot box.

Question 14. Does this verification require the intervention of an additional party if the voter cannot read (because of blindness or lack of education)?

Some handicapped people favor e-voting machines because they can be programmed to help those with visual impairments ([47], p. 301) and oppose voter-verifiable audit trails because "they are inherently inaccessible to the handicapped" ([25], pp. 84–85). However, e-voting machines can be equipped with audio units that will serve the same purpose as paper audit trails with respect tot he voters ([33], p. 11). These do not serve the audit function that paper trails serve, however; see below.

Bederson and Herrnson noted severe human interface problems with the audio system used; specifically, "it is hard to navigate, it is difficult to have questions repeated, the audio quality is poor, the buttons provide no feedback, there is no opportunity to review the ballot" ([2], p. 599).

The question of how a visually-impaired or illiterate person could validate his or her vote is a key question to enabling those voters to vote. The information about how vendors handle these voters, and votes, is vague.

Question 15. What requirements is your e-voting system designed to meet?

There are many requirements that e-voting systems must meet. These requirements are of two kinds. First, they must satisfy the fundamental or legal requirements for election, such as not counting overvotes, counting votes accurately, and so forth, that are common to all elections. Sec-

ond, there are requirements particular to the e-voting systems, such as requiring validation of updates. The former are typically encoded into laws and regulations. The latter may vary from jurisdiction to jurisdiction and vendor to vendor. Examples of the former include the requirement that, beginning in 2006, e-voting systems notify users of overvotes and allow them to fix their ballot before it is cast ([18], p. 554); examples of the latter include that third party components be documented ([25], p. 48), that there be redundant storage of votes ([27], p. 173), and that supervisor functions be protected by passwords ([25], p. 95).

A common problem is that vendors and election officials discuss the first type of requirement but rarely provide detailed information about the second type. For example, in [51], Williams states that systems are tested to "verify that the voting system complies with the requirements of the Georgia Election Code, the Rules of the Georgia Secretary of State, and the Rules of the Georgia Election Board." These are legal requirements. He then states that testing at the county offices verifies that "the system … is working properly." Without further elaboration, it appears he means that it meets the legal requirements, and Georgia relies on others to check the requirements particular to e-voting systems. The comment "... and is identical to the system that was previously Qualified by the ITA and Certified by the State" supports this conclusion.

The key regulations for e-voting systems come from the Federal Election Commission's *Performance and Test Standards for Punchcard, Marksense, and Direct Recording Electronic Voting Systems*. Although these are voluntary, many states require that vendors meet these standards ([25], p. 154). Further, these standards appear to be minimal and issued in response to problems in the 1980s; they "fail to cover many issues" ([27], p. 154) and have just been revised. The testing laboratory may test to these standards, as well as others. For example, comments the Johns Hopkins researchers found in the analyzed code suggest that Wyle Laboratories, which tests e-voting systems, has its own set of requirements for the systems ("… evidence … comes from a single log comment: 'Modify code to avoid multiple exit points to meet Wyle requirements.' This could refer to Wyle Laboratories whose website claims that they provide all manner of testing services." [29], p. 212; Jones confirms that "only Wyle Labs of Huntsville Alabama is available as an independent testing authority." [27], p. 154).

Vendors should specify *exactly* what requirements their e-voting systems meet. If a vendor cannot do this, then the e-voting system cannot be tested to see if it meets those requirements. Any additional requirements, such as those of testing and validation authorities, should also be known to both the vendors and the purchasers.

Further, the requirements should be meaningful. A common type of requirement is that "event X should occur no more frequently than 1 time in 1,000,000 events." This requirement is meaningless for two reasons. First, why was the "1 time in 1,000,000" chosen? Why not "1 time in 100,000" or "1 time in 10,000,000"? Without justification, the need for the requirement is unclear, as is the benefit of meeting it. The second problem is how the "events" are measured. If an event occurs, a tester decides to cancel (undo) the event, and then redo it, is that 1 event or 2? If "event" is not defined precisely, how will the tester know when it occurs? For example, if an event is "a pull of the level", has an event occurred if someone pulls the lever halfway rather than all the way? Meaningful requirements are an unambiguous as possible. The above requirement is quite nebulous, and ambiguous.

Currently, the literature surveyed did not offer lists of detailed requirements used by vendors. For example, one vendors stated that its "software is under constant development and improvement to meet customer needs and ever-changing statutory requirements" ([12], p. 262),

and all vendors refer to the statutory and FEC requirements. An election officials also sid that "the system satisfies the requirements of RCW 29.33.300" ([41], p. 183). While laudable, these requirements are not specific enough to develop programs from; they must be further refined to specific statements of the form "the system will perform XXX in one of the following ways" or "the system will not do XXX" or something similar. These statements of requirements can then be used to check the programs and systems directly.

Question 16. How do you know it meets them?

Given that the requirements that the e-voting system must meet are specific and meaningful standards, a vendor should be able to describe exactly how the e-voting system was tested, and present the results. The results should specifically state not simply what was validated, but also what was *not* validated. The reason for this lies in the nature of testing.

A common belief is that testing proves a program or system works without error. Testing proves no such thing. Testing can demonstrate the presence of errors; it cannot demonstrate correctness. An example may help clarify this.

Suppose I am selling e-voting equipment, and I contract with someone else to provide the routines that "talk" to the hardware used to store the votes. I need special software to do this because I am replicating the votes on 3 different devices, and want that to happen automatically. The routine provider wants to create confusion in the election, so he writes his software routines to arbitrarily change every tenth vote to the first candidate on the ballot, providing the election is on the first Tuesday in November. As a vendor, I run tests on the e-voting system. Unless it is the first Tuesday in November, my tests will not reveal this attack. Further, if the software routine writer is any good, he will make the driver delete the part changing the votes when the system is turned on any day after the first Tuesday in November.

One can be considerably more subtle. Ken Thompson, one of the inventors of the UNIX operating system, described how to insert trap doors into software in such a way that checking source code would not detect the trap door, nor would rebuilding the software eliminate the trap door [48][7]. Other methods of attack involve modifying the firmware, hardware, third-party software, or operating systems of computer systems used to build e-voting systems. The question is how far the attacker will go to subvert the system.

These examples raise the question of whether *any* system can be proven to work correctly. In principle, by the application of mathematical methods, yes; however, reality is very different. Mathematical methods (called "formal methods" in the literature) rely on systems to do the mathematics, and these in turn must be validated. This raises the same questions as before. So, in practise, no system can be proven to work correctly. Instead, one tries to validate the system to show it functions without error in a reasonable set of cases, "reasonable" being defined by the purpose to which the system will be put.

The usual way to demonstrate adequacy (as opposed to "correctness") is to apply standards. There are two basic types of standards: those that focus on the *process* of developing systems, and those that focus on the *result* of developing those systems. The former types of standards include the ISO 9000 family, which verifies that quality assurance (*not* development procedures). According to the InfoSENTRY report, Diebold's manufacturing facilities meet these standards ([24], p. 492)[8], as do some of ES&S's facilities ([24], p. 495), Hart InterCivic and its

7. Karger and Schell [28] proposed this attack in 1974.

manufacturing partners ([24], p. 497), and Sequoia's manufacturing firm ([24], p. 499). This type does not assure that the end product is accurate or reliable, but does insure that the steps producing that end product are documented and tested.

The second type of standards include the ones put forth in Federal Election Commission's *Performance and Test Standards for Punchcard, Marksense, and Direct Recording Electronic Voting Systems*, and by various secretaries of state (see for example [41]). The process by which systems are constructed to meet these standards is irrelevant; the testing is performed against the end result, and adequacy is determined by the results of the testing. It is important to understand that the standards do *not* assure accuracy; they are designed to catch gross inaccuracy *only*. They do not cover the full range of software and hardware development for e-voting systems. Software that passes tests designed to measure conformance to these standards can still be compromised using techniques such as the Karger and Schell method Thompson used. The second problem is that the developers need give no assurance other than passing tests that the systems function correctly. Given the complexity of software development, and indeed of the resulting software, it is entirely possible for a malicious entity to evade detection during testing. In theory, no amount of testing could uncover certain problems; in practise, the problems that testing would uncover would be obvious ones.

Therefore, vendors should meet both types of standards. This again does not assure accuracy, but gives more confidence than a vendor whose product meets only development standards and not end-result standards, or vice versa. Further, the current testing is clearly inadequate. It does not test the underlying operating system, as pointed out earlier (see the discussion under Question 1). It also does not test the e-voting system adequately.

Vendors have described the current testing process as rigorous: "the software goes through rigorous testing and certification by one of three companies … [t]hose companies 'go through every line of code'", according to Mr. Swidarski of Diebold ([50], p. 5). But Jones points out that third parties have breached administrative security on certified e-voting systems because of astoundingly obvious problems that the certification testing simply does not check for. The example cited if particularly egregious. When examining Diebold's system, InfoSENTRY found Diebold using a PIN (password) for a supervisor smart card that was the same for *all* such cards, was set to something so obvious it could be guessed in 3 tries, and could only be changed in the factory making the cards ([25], p. 95). Each of these violates basic principles of secure design. Williams reports that e-voting stations in Georgia undergo Logic and Accuracy tests to confirm that the e-voting stations are correctly configured and working before they are sent to the precincts. Also, a test center at Kennesaw State University runs mock elections to assure that the e-voting system can handle the maximum number of ballots that can be cast at a Georgia precinct. ([51], p. 107) But the effectiveness of this testing was called into question in an interview done by Harris with a technician checking Georgia e-voting machines. Approximately 25% of the e-voting systems running the certified software failed to operate correctly, possibly due to an operating system problem ([39], p. 138). The systems were upgraded, but the fixes apparently had not been certified ([39], p. 139). The technician's discussion of the Logic and Accuracy testing describes a non-rigorous testing procedure in which one vote was entered ([39], p. 140). It is unclear if this was

8. Diebold's comments that their software is constantly being updated (as quoted in [19], p. 226) appears to indicate that the ISO 9000 process they use is not as rigorous as it should be, or that their process was changed after their comments to conform to ISO 9000. The process described in [39], for example, would not comply with an ISO 9000 certified quality assurance program.

the official Logic and Accuracy test, or if the system was later tested officially by Kennesaw State University, and if the latter whether one system, some systems, or all systems were tested.

The SAIC analysis of the Maryland voting systems raises an interesting point. The report recommends "including testing for time-triggered exploits (e.g., Trojans [Trojan horses]) as a part of the L&A testing. If L&A testing proves to be an inappropriate venue for this testing, we recommend the [State Board of Elections] choose another venue, or introduce into the testing protocol an additional battery of tests including these procedures" ([43], p. 324). The implication is that none of the testing protocols used require testing for these attacks. [9]Again, testing for these attacks is basic to any credible security testing methodology.

The InfoSENTRY report also raises questions about the effectiveness of the testing done on e-voting systems ([5], p. 476). From the InfoSENTRY report [24], it is clear that these systems had been certified for use. They found high risk areas of concern in the e-voting systems of four vendors (out of four vendors whose systems were tested).

That subtle problems undetected by certification testing are not fantasy is clear from events in Fairfax County, Virginia. In the November 2003 election, some voters complained that their mark for School Board candidate Rita Thompson disappeared a few seconds after they made it, and county officials discovered that one of the e-voting machines seemed to subtract one vote every hundred for that candidate. As the race was close (her 77,796 votes left her 1,662 votes behind the winner), this bug could have affected the results of the election ([8], p. 33–34).

This also points out that conditions under which testing is performed can affect the results of the testing. The systems used at Fairfax had been tested, both in the laboratory and by poll workers, but had not been put through the rigors of tests mirroring actual use in an election. Such testing would be more likely to have detected the problem than other types of testing.

Vendors should be asked to describe the testing that they, or others, have done *beyond* the certification standards of both the federal and state governments. Further, the vendor should be prepared to describe the methodology and show the results, and others should be able to reproduce these results. The reason is that assurance is evidence that convinces one that a system meets its requirements, and evidence that is not available for review and analysis by outsiders is far less convincing that evidence that outsiders can study, review, and reproduce in their own laboratories. Testing should include laboratory and certification testing, and testing under conditions identical to those under which the system will be used.

Finally, an obvious question is whether the software used in the tests is the same software that is deployed to the e-voting stations in precincts. In the November election in California, Diebold apparently uploaded code to upgrade the software in several e-voting machines in Alameda County—and the uploaded code was not certified by the state, as required by state law ([16], p. 17). A quick investigation showed that the e-voting systems in 17 counties were not certified by the state,[10] and two counties used systems not certified by the federal government. The vendor,

---

9. The April 2003 Federal Election Commission's Voting System Standards require that vendors "shall deploy protection against the many forms of threats to which they may be exposed such as file and macro viruses, worms, Trojan horses, and logic bombs" ([15], Volume 1, Section 6.4.2, p. 6-7). The wording suggests that the threat is from software *external* to the voting software (such as bogus smart cards). There appears to be no requirement that the e-voting software itself be checked for Trojan horses.
10. At the time, the latest version of the GEMS software that California had certified was 117.17; the counties were using versions 117.20, 117.22, 117.23, 118.18, and 118.18.02 ([25], p. 96).

Diebold, admitted being "negligent in notifying the state about changes in its software … [but the] changes in the software had been 'cosmetic'" ([1], p. 531). Of course, changes in certified software invalidates the current certification, because they may affect the ability of the modified software to meet the certification requirements.[11] The system must be recertified.

Vendors should describe how they handle the recertification and re-testing (meaning tests beyond those required for certification) of modifications to their systems. This includes the underlying software, such as the operating system. Any vendor who states that they check only when the modification will affect the voting software when the new software is to be deployed should be asked how it makes that determination, and why it does not perform testing and certification upon the changed software. (A vendor may make in-house modifications to systems that are not deployed. Those are irrelevant. But any system that has changed since last certified or tested should undergo recertification and re-testing before being deployed.)

Question 17. Once an e-voting system is in the field, how do you handle updating the software and hardware on that system?

Updating the software on a system changes the software. The goal is to have the new software installed on the system, presumably to improve some aspect of the system's function or assurance. The threat is that the new upgrade may not do this, may be mis-installed so the system is not improved, or a fake upgrade is installed and corrupts the system.

The issues here are how the vendor assures that the right upgrade is obtained, and how it is installed. Diebold has placed patches on *FTP* servers that customers could download ([39], p. 138); from the description, it is unlikely that the connection was authenticated or the patch was integrity-checked. This means that the recipients could not check that the upgrade code was from the vendor and had not been modified since it was placed on the *FTP* server. Also, the vendor distributed upgrades to the operating system as well as to the GEMS software ([39], p. 138).

In California, at least, some upgrades had not been certified to meet state requirements ([1], p. 531; [16], p. 15, reporting allegations; [23], p. 529; [32], p. 581).

There is little information on the nature of the personnel performing the upgrades; it indicates that the personnel are competent, if overworked, and that in some cases the process is chaotic (see [39]). This highlights a tension described by Fischer ([18], p. 551) in the need to take time to follow maintenance and upgrade procedures conflicting with the need to hold a timely election.

Vendors should ensure that all upgrades are certified as required by law (federal, state, and any other applicable statutes) before installation. At a minimum, the vendors should cryptographically sign upgrades so the installers can check that the upgrade originated with the vendor and was not modified. Further, only qualified support personnel should perform the upgrades.

Question 18. What do maintenance personnel do when they work with the e-voting systems?

---

11. A (possibly apocryphal) story may help clarify this point. A computer system was rated B2 under a set of standards called the "Trusted Computer Security Evaluation Criteria" (also called the Orange Book) [10] developed by the National Security Agency and widely used for many years. A B2 rating provided substantial security, including rigid requirements on protecting passwords. The vendor made some minor modifications and upgraded the system. The "minor modification" accidentally made all user passwords on the system available to anyone, thereby eviscerating the security of the system. The "minor modification" changed a B2 system into an unrated system with an obvious security flaw.

Access by maintenance people can be controlled in a number of ways. Diebold's GEMS uses a special smart card ([29], p. 201) and a password (PIN). The Hopkins study pointed out that an attacker could easily guess the password ([29], p. 202); the InfoSENTRY study also pointed out the PIN was easy to guess and could only be changed in the factory ([25], p. 95). Others ([37], p. 219; [42], p. 352; [18], p. 549) state the type of attack suggested would fail because of procedural controls.

Another interesting question is the administration of the election server, which holds the database used to tally the precinct results. Diebold's GEMS system uses Microsoft Access® running on a Microsoft Windows 2000 system ([11], p. 264). Security problems with all versions of Windows are widely known; for this reason, it is *imperative* that the election server be physically disconnected from all networks (including telephone lines) except when election results are uploaded, and even then only when the connection can be verified. An effective protocol is to have the precinct call in and ask that a modem be activated; the results are uploaded, and then the modem is disconnected. This isolates the system except when actual election results are uploaded. Of course, only trusted users (and administrators) should be allowed to access this system. Marsh [35] discusses how to compromise an election server connected to a network; the same mechanisms will work if the users of the system are not trusted.

Question 19. If the system is upgraded or has maintenance work done, how can we verify that it still tallies votes accurately, and meets our other requirements?

This question acknowledges the reality that, once a certified system is modified, it is no longer the same the certified one. Vendors should recertify and retest the modified systems completely, and when upgrades are distributed, require that the system to be upgraded is the same system that the upgrade was developed for before the upgrade is performed. For example, if upgrade X2-3 upgrades a version 2 e-voting system from E-Voting, Inc. to a version 3 e-voting system, the upgrade should *not* be run on a version 1 e-voting system from E-Voting, Inc.

Currently, upgrading a system requires either that the entire system be recertified, or the upgrade be recertified. As noted, some vendors have been lax about this. Maintenance work (testing components of the system, etc.) is not discussed at all in the literature.

Question 20. What audit mechanism, or external vote tally record, does your system supply, and how do you know it is correct?

The key here is *external* to the e-voting system. The problem with an internal mechanism is that it is produced by the system being audited, and there is no independent verification of the data. In non-technical terms, an internal audit mechanism is like an auditor asking the treasurer of a company if the books are in order, being told yes, and when the auditor is asked to see the books, the treasurer replies that the books are all kept mentally, so he can write them out for the auditor. If the treasurer is corrupt, the auditor probably will not detect it from what the treasurer writes. Similarly, an internal vote tally merely confirms what the system said earlier, unless the system is completely corrupted. An external audit trail allows the checking of the votes recorded in the system against something that each voter has checked (or had the opportunity to check), and so will detect more corruption than the internal trail. To return to our non-technical example, it is equivalent to the books of the company being on paper, and the auditor can confirm the transactions in the company files against the numbers in the books. Corruption is still possible, but it is a lot harder to escape detection.

Many states require such mechanisms ([22][40][46]). Some states do not. Williams ([51], p. 114) contends that the use of such a trail is unmanageable.

The Diebold AccuVote-TS voting system has an internal ribbon printer, though ([12], p. [43], p. 333); presumably this could be used to generate the audit trail, much as an ATM can generate a receipt.

Question 21. If an auditor needs to validate your claims of accuracy, how could he or she use the audit mechanism, or external vote tally record, to verify your claims?

Validation with an external audit mechanism as described above is straightforward: tally the votes and compare them to the machine-generated results. The external audit trail should identify the actual e-voting system on which the ballot was cast, to aid in identifying faulty systems.

If the audit trail is generated at the end of the election by the e-voting machine, it *cannot* be used to validate a claim of accuracy. When this occurs, the e-voting system is simply printing out the votes it has already recorded—and if it made errors recording those votes, those errors will persist in this audit trail. Having a voter-verifiable trail, so a voter can check that his or her vote was correctly recorded, is the *only* way to ensure a meaningful audit of the votes. As votes are secret, any other auditor would be unable to validate the audit trail against the actual votes because she would not know the actual votes.

This supports the need for a voter verifiable audit trail. Vendors must provide this to allow the accuracy of the systems to be checked in an election.

Conclusion

Election returns must be accurate and verifiable. If they are not accurate, the wrong people or propositions may be declared the winner, undermining confidence in the effectiveness of the public's mechanism for controlling government. If they are not verifiable, the results may be *perceived* as inaccurate, which will cause the same disaffection as an inaccurate result. Further, non-verifiable results imply the election officials and equipment will not make mistakes that will create an inaccurate result. Over the years, election officials have evolved a number of mechanisms to ensure that elections have these two qualities. Let us focus on two in particular.

The first mechanism corresponds to *separation of privilege*, a concept in security that states if one component fails, a second will be in place to correct the failure. Elections use this principle in many places. For example, in recounts, one person does not recount the ballots by hand. At least two people are involved, and they are drawn from disinterested parties (such as non-partisan groups) or from competing parties (for example, a Democrat and a Republican will observe the counter). The use of electronic systems does not change this. In California, for example, the state legislature mandated that, if electronic ballot counting machines are used, the ballots of at least 1% of the precincts in the county must be counted by hand, and the totals checked against those provided by the machine. This is intended to catch problems with the electronic ballot counting systems.

The second mechanism corresponds to *reproducibility*, a concept in security that states that a result can be independently reproduced to verify that it is correct. (This is also a fundamental principle of science. Something that cannot be reproduced is not accepted as valid.) Here, the ability to reproduce the results from the raw data (cast ballots) will verify the integrity of the original results (or correct them if they are wrong). Indeed, problems arise when the raw data allows

disputes among the people doing the recount, as the debacle in Florida in the 2000 Presidential election showed.

Consider how these mechanisms can be applied to e-voting systems.

If an e-voting system fails, there must be some way to correct the problem. If the failure is a system crash, other systems could replace it. But if the system incorrectly records votes, the election officials and voters can realize this only if the reported results are clearly erroneous (such as the e-voting system in Boone County, Indiana, that reported over 144,000 votes cast in by under 6,000 voters [31] or the system in Broward County, Florida, that failed to report about 70,000 votes [3]) or if there is a physical record of the votes cast. By the nature of electronic systems, if the vote is mis-recorded, it will remain mis-recorded in memory, and the election officials cannot determine if the recorded vote is the one the voter intended to cast. But the voter verifying his or her vote by examining the media on which it is recorded overcomes this objection, because the voter can change the vote if it is incorrect (by requesting a new ballot, for example, and destroying the old one in front of the election officials). Of course, a voter cannot look at the innards of the e-voting system to do this. But a voter-verifiable audit trail (which need not be on paper *for this purpose*) will enable the voter to check the vote cast by examining a representation of that vote. This applies the mechanism of separation of privilege: if the e-voting system fails, the audit trail will indicate an erroneous vote, and the voter can decline to verify the recorded vote.

The need for reproducibility means that a record of the original votes cast (the "raw data") must be available. The question is what the "raw data"? For an e-voting system, it is the input to the e-voting system, because that reflects how the voter expressed his or her votes. The e-voting system processes this data into its internal representation. So, working from the data on the e-voting system is not working with the "raw data" of the election. Instead, some "post-election audit trail" built from the raw data itself must be available. The obvious way is to use the voter-verified audit trail described above, because then the voter has checked the results in that trail. This, incidentally, suggests that the voter verifiable audit trail must have a paper representation, so the post-election analysis (should there be one) can use the data from that trail. The use of e-voting systems has an advantage over punch systems, because the audit trail will not suffer from ambiguity; "hanging chad" is not a problem when the results are printed. These ballots can be recounted with a very high degree of accuracy, assuming the print is readable.

One unresolved question is whether the post-election audit trail should be used to compute the "official" election results. The reason for this question arising lies in the nature of the e-voting systems' representation of votes. Suppose the system makes an error that cannot be detected at once. If no-one calls for a recount, will the error be detected? Perhaps something like what the California legislature did for electronic ballot counting systems: count at least 1% of the ballots for every precinct, and compare the totals to those produced by the relevant e-voting systems. Such a preventative mechanism should be discussed.

Electronic voting systems have far to go before they can supplant paper. The key issues are separation of privilege and reproducibility. To say that election procedures are sufficient to catch errors in e-voting systems is to make a number of assumptions not warranted by the methods used to produce and test the technology. If election procedures are lax, e-voting systems can commit gross errors without detection. If election procedures are meticulous, e-voting systems can commit subtle errors without detection. A voter-verified audit trail, and a post-election audit

trail, provide additional assurances. But they also raise the question of when the latter will be used.

Reference

[1] E. Ackerman, "Voting machine maker dinged", *San Jose Mercury News* (Dec. 17, 2003); *item 5–61*.

[2] B. Bederson and P. Herrnson, "Usability Review of the Diebold DRE System for Four Counties in the State of Maryland" (2003); *item 6–1*.

[3] E. Benn, "Broward vote total off in reporting glitch", *Miami Herald* (Nov. 6. 2002); *item 3–7*.

[4] M. Bishop, *Computer Security: Art and Science*, Addison-Wesley Publishing Co., Boston, MA (2002).

[5] K. Blackwell, "Blackwell Seeks Improvements and Additional Security Assurances from Electronic Voting Machine Vendors", Office of the Secretary of State of Ohio (Dec. 2, 2003); *item 5–57*.

[6] J. Boyer, "Inferring Sequences Produced by Pseudo-Random Number Generators", *Journal of the ACM* **36**(1) pp. 129–141 (Jan. 1989).

[7] J. Boyer, "Inferring Sequences Produced by a Linear Congruential Generator Missing Low-Order Bits," *Journal of Cryptography* **1**(3) pp. 177–184 (1989).

[8] D. Cho, "Fairfax Judge Orders Logs Of Voting Machines Inspected", *Washington Post* p. B01 (Nov. 6, 2003); *item 3–8*.

[9] D. Cho and L. Rein, "Fairfax To Probe Voting Machines", *Washington Post* p. B01 (Nov. 18, 2003); *item 3–9*.

[10] Department of Defense, *Trusted Computer System Evaluation Criteria*, DOD 5200.28-STD (Dec. 1985).

[11] Diebold Election Systems, "AccuVote-TS™"; *item 5–21*.

[12] Diebold Election Systems, "Checks and balances in elections equipment and procedures prevent alleged fraud scenarios" (July 30, 2003); *item 5–20*.

[13] Diebold Election Systems, "Summary technical analysis of recent voting system report" (July 29, 2003); *item 5–18*.

[14] D. Dill, "Commentary on the SAIC Report: 'Risk Assessment Report: Diebold AccuVote-TS Voting System and Processes'", *VerifiedVoting.org* (Sep. 26, 2003); *item 5–56.*

[15] Federal Election Commission, "Voting System Standards" (Apr. 30, 2003), at http://www.fec.gov/pages/vssfinal/vss.html

[16] P. Festa, "California voting machine called into question," *news.com* (Nov. 4, 2003); *item 3–1*.

[17] Financial Times, "Virus Hts A.T.M.'s and Computers Across Globe", *The New York Times* (Jan. 26, 2003); available at http://www.nytimes.com/2003/01/26/business/FT1042491212045.html?ex=1073365200&en=c48e10b6b05147e0&ei=5070.

[18] E. Fischer, “CRS Report for Congress: Election Reform and Electronic Voting Systems (DREs): Analysis of Security Issues”, Congressional Research Service, The Library of Congress, Washington DC (Nov. 2003); *item 5–62*.

[19] B. Harris, “Diebold Denies Ease of Voting Machine Tampering—But Rebuttals Don’t Stand Up”, *Scoop* (Jul. 2003); *item 5–17*.

[20] B. Harris, “U. S. Election Integrity Flaw Discovered At Diebold”, *Scoop* (Feb. 10, 2003); *item 5–3*.

[21] B. Harris, “Voting System Integrity Flaw; System Integrity Flaw Discovered At Diebold Election Systems”, *Scoop* (Feb. 5, 2003); *item 5–2*.

[22] D. Heller, “Secretary of State Heller Announces Direct Recording Electronic Voting Machine Choice” (Dec. 10, 2003); *item 5–67*.

[23] A. Hoffman and T. Reiterman, “Secretary of State Orders Audit of All Counties’ Voting Systems”, *Los Angeles Times* (Nov. 13, 2003); *item 5–61A*.

[24] InfoSentry Services, Inc., “Volume 1, Computerized Voting Systems Security Assessment: Summary of Findings and Recommendations” (Nov. 21, 2003); *item 5–58*.

[25] D. Jones, “The Case of the Diebold FTP Site” (2003); *item 5–1*.

[26] D. Jones, “E-Voting -- Prospects and Problems” (Apr. 2000); *item 5–54*.

[27] D. Jones, “Problems with Voting Systems and the Applicable Standards”, testimony before the House of Representatives’ Committee on Science, Washington DC (May 2001); *item 5–9*.

[28] P. Karger and R. Schell, “MULTICS Security Evaluation, Volume II: Vulnerability Analysis,” ESD-TR-74-193, Vol. II, Electronic Systems Division, Air Force Systems Command, Hanscom Field, Bedford, MA (June 1974).

[29] T. Kohno, A. Stubblefield, A. Rubin, and D. Wallach, “Analysis of an Electronic Voting System”, Technical Report TR-2003-19, Information Security Institute, Johns Hopkins University, Baltimore, MD (July 23, 2003); *item 5–13*.

[30] T. Kohno, A. Stubblefield, A. Rubin, and D. Wallach, “Response to Diebold’s Technical Analysis, (Jul. 2003); *item 5–22*.

[31] R. Konrad, “Apparent security hole in primary highlights danger of electronic voting,” *San Diego Union-Tribune* (Sep. 10, 2003); *item 3–3*.

[32] D. Kucinich, “Voting Rights” (undated); *item 5-66*.

[33] J. Legon, “Electronic elections: What about security?” *CNN* (Nov. 5, 2002); *item 2–3*.

[34] A. McFeatters, “Computer voting viewed skeptically,” *Post-Gazette* (Aug. 7, 2003); *item 5–27*.

[35] J. Marsh, “Diebold’s Vote-Tally Software—Security Review Instructions”, Version 5.0i (Sep. 17, 2003); *item 5–37*.

[36] R. Mercuri, “A Better Ballot Box?”, *IEEE Spectrum* **39**(10) pp. 46–50 (Oct. 2002).

[37] R. Mercuri, “Critique of ‘Analysis of an Electronic Voting System’ Document” (July 2003); *item 5–14*.

[38] P. Neumann, “Security Criteria for Electronic Voting”, *Proceedings of the 16th National Computer Security Conference*, Baltimore, MD (Sep. 1993).

[39] D. Paull, “[IP] Interview with Georgia Diebold Election Machine installer”, *interesting-people message* (Aug. 8, 2003); *item 5–8*.

[40] S. Reed, “Reed requires paper audit trail” (Dec. 16, 2003); *item 5–68*.

[41] S. Reed, “Report of the Secretary of State on the Examination and Evaluation of an [*sic*] Direct Recording Electronic Vote Tallying System”, Office of the Secretary of State of Washington (Sep. 6, 2002); *item 5-10*.

[42] SAIC, “Appendix B: Security Statements from the Rubin Report & State of Maryland Controls” (Sep. 2003); *item 5–32*.

[43] SAIC, “Risk Assessment Report: Diebold AccuVote-TS Voting System and Processes” (Sep. 2003); *item 5–31*.

[44] R. Saltman, “Accuracy, Integrity, and Security in Computerized Vote-Tallying”, NBS Special Publication 500-158, Institute for Computer Sciences and Technology, National Bureau of Standards, Gaithersburg, MD (Aug. 1988).

[45] J. Saltzer and M. Schroeder, “The Protection of Information in Computer Systems”, *Proceedings of the IEEE* **63**(9) pp. 1278–1308 (Sep. 1975).

[46] K. Shelley, “Secretary of State Kevin Shelly Announces Directives To Ensure Voter Confidence in Electronic Systems” (Nov. 21, 2003); *item 5–68A*.

[47] E. Smith, “E-voting becomes touchy topic”, *Akron Beacon Journal* (Aug. 15, 2003); *item 5–29*.

[48] K. Thompson, “Reflections on Trusting Trust,” *Communications of the ACM* **27** (8), pp. 761–763 (Aug. 1984).

[49] VoteHere Press, “Sequoia Voting Systems and VoteHere to Provide Additional Electronic Ballot Verification Options for AVC Edge Touch Screen Voting System” (Aug. 4, 2003); *item 5–46*.

[50] M. Warner, “Machine Politics in the Digital Age”, *The New York Times* (Nov. 9, 2003); *item 1–1*.

[51] B. Williams, “Security in the Georgia Voting System” (Apr. 2003); *item 5–4*.

[52] L. Witt, “More Calls to Vet Voting Machines”, *Wired News* (Aug. 2003); *item 5–28*.

[53] K. Zetter, “E-Vote Machines Face Audit”, *Wired News* (Aug. 2003); *item 5–59*.