# Analysis of Volume Testing of the AccuVote TSx / AccuView

**Matt Bishop, Loretta Guarino, David Jefferson, David Wagner**
**Voting Systems Technology Assessment Advisory Board**
with assistance from statistician Michael Orkin (Managing Scientist, Exponent)
October 11, 2005

## Executive Summary

In this report we analyze the results of the recent volume testing conducted in Stockton on July 20, 2005 of 96 Diebold AccuVote TSx machines.

34 to 36 failures were recorded during the test. The count is a little ambiguous because at least one printer incident may have been recorded twice, and one incident, the failure of the printer housing to latch the first time, might be considered too minor to include. To be conservative, we have assumed the lower failure count, i.e. 34 incidents (14 printer problems and 20 software crashes). The data from which we worked, derived from the hand written incident reports, is in an associated spreadsheet.

Our analysis suggests that this failure rate could be serious, especially given the preponderance of software crashes. From the data we estimate the Mean Time Between Failures (MTBF) of these machines to be approximately 15 hours under the conditions experienced during volume testing. It is unclear what failure rate this might imply for a real election.

We found the many software failures potentially more troubling than the paper jams. It seems likely that further changes to the AccuVote TSx software will be required.

Under one possible interpretation of the standards, the failure rate observed during these tests was more than 10 times higher than permitted by federal standards (which require a 163-hour MTBF). The failure to detect this fact during the ITA's testing process appears to be due to serious defects in the testing methodology specified by federal standards. One lesson of this analysis is that the testing performed during the federal qualification process is apparently inadequate to ensure that voting machines will be reliable enough for use in elections.

## Introduction

On July 20, 2005, 96 Diebold TSx DREs with AccuView printers machines were tested by the Secretary of State's office over a period of 5.33 hours in a setting designed to emulate a real election. This appears to be a first: as far as we are aware, no controlled test of this scale has ever been performed before anywhere. Thus, this provides an opportunity not only to assess the reliability of the TSx, but also to examine the effectiveness of volume testing in general.

This report is in three parts: (a) we analyze the data collected during the volume test, and draw conclusions about the reliability of the TSx; (b) we discuss the implications for how to recover from failures; (c) we evaluate several possible methods that the State of California could use to enhance reliability testing of voting machines.

### Analysis of the TSx Volume Test

The volume test was performed as follows. First, 96 TSx machines were set up two days in advance and "burned in," and local election officials and contractors were invited to help test them. On July 20,

approximately 48 people showed up and tested the machines from 9:00am to 4:00pm. The machines were in use for a total of 5.33 hours, after subtracting for several breaks during which the machines were not used. Each tester was assigned two machines. Each failure encountered was recorded by one of three recorders. Recording was not uniform; a typical entry recorded some subset of the machine number, machine serial number, time of failure, type of failure, method used to remediate the failure, ballot number when the failure occurred. However, few entries contained all of this information. For some of the machines, the recorders also provided us with the final count of the number of ballot images recorded electronically and/or the number of VVPAT images printed.

To simplify analysis, we first transcribed this data into a uniform format, with one row for each recorded failure. A spreadsheet is appended to this report containing the data we used.

**Overview of results.** We found that there were 34 failures, spread across 29 distinct machines. We classified each failure into one of two categories: (a) printer jams, and (b) software failures, where the touchscreen machine crashed, froze, hung, or reported an unrecoverable error condition. The 34 failures broke down into 14 printer jams and 20 software failures, with 12 machines experiencing at least one printer jam (2 machines suffered from 2 printer jams) and 18 machines experiencing at least one software failure (2 machines encountered 2 software failures). One machine experienced both a printer jam and a software failure. We discuss both printer jams and software failures in further detail below.

One way to characterize the rate at which failures occur is by calculating the Mean Time Between Failures (MTBF). This is the metric that is used in federal standards. However, depending upon the cause of failures, the failure rate might vary depending upon the environmental conditions and how heavily the voting equipment is used, so the MTBF metric must be interpreted with caution. Another possible metric is the Mean number of Votes Between Failures (MVBF). The latter metric would be more appropriate if the number of failures is expected to vary directly with the number of votes cast, rather than the amount of time that the machine is turned on. Therefore, we calculate both metrics.

Another issue is that software failures are potentially more serious than printer failures. All of the printer failures encountered in the volume test were recoverable. In contrast, as we discuss later, software failures can potentially have a more severe impact. Therefore, we analyze printer and software failures separately. Because we expect the number of printer failures to be proportional to the number of votes cast, not the number of hours the equipment is in use, the MVBF is likely to be a much better measure of printer failures than the MTBF.

Working from the data collected during volume testing, we estimated both the MTBF and MVBF of the combined TSx/AccuView unit. When considering only software failures, the estimated MTBF is approximately 25.6 hours. This means we expect a typical TSx machine to experience a software failure about once every 25.6 hours, under the conditions experienced during the volume testing. The estimated MVBF, when considering software failures only, is approximately 536 votes, meaning that one might expect a software failure about once every 536 votes. The mathematical computations are given later.

When all failures are included, our computations indicate that the MTBF is approximately 15 hours, under the conditions experienced during the volume test, and the MVBF is 315 votes. Again: we caution that the MTBF may be misleading when printer failures are included; we mention the aggregate MTBF only because it is the metric used by federal standards.

This failure rate has consequences for the availability of these machines. During a typical election, the polls are open for 13 hours. If the conditions during a real election were comparable to the conditions during the volume test, then we could use the MTBF to estimate the number of failures likely to be observed during an election. For instance, if we assume that we can recover from printer jams, but

machines are taken out of service upon any software failures, then such calculations would suggest that almost 40% of machines would experience a software failure and need to be taken out of service, leaving only 60% of machines in working order by the close of polls. However, in practice this is likely to overestimate the number of failures. During the volume testing, the machines were in heavy use throughout the 5.33 hour testing period. In contrast, in a real electionmachines might be idle for a longer period of time, and it is likely that votes would be cast at a slower rate during a real election.

The MVBF is likely to yield a better predictor of the frequency of software failures. It seems more likely that TSx failures may occur at a rate that is directly proportional to the number of votes cast, not the number of hours in operation. During the volume test, an average of 111 votes were cast per TSx machine. If each TSx machine received the same number of votes in a real election, then one could expect to see a similar fraction of machines failing during election day as observed during the volume test. During the volume test, approximately 20% of machines experienced a software failure, so one might expect roughly 20% of machines to experience a software failure and need to be taken out of service during an election of comparable scale.

Under these assumptions, some polling places would be left without any working machine by the end of the day. Obviously when the failure rate is this high, recovery from failures is a critical issue. We return to recovery issues later. The observed failure rate appears to be far larger than the MTBF called for in the relevantfederal standards. Both the 1990 and 2002 FEC standards require a MTBF of at least 163 hours. On the surface, then, the aggregate failure rate observed during the volume test would appear to be more than 10 times higher than permitted.

That said, it is not entirely clear how to compare the observed MTBF rates with the requirements in the FEC standards. The FEC standards specify a particular measurement methodology: the MTBF is to be calculated by counting the number of failures observed as the ITA performs its test duties. Unfortunately, the FEC's methodology is quite ambiguous and seems to do a poor job of mimicing real-world election conditions. In comparison, the volume tests do a much better job of evaluating voting equipment under realistic conditions, because volume testing involves live humans casting votes in an environment intended to simulate an election. Therefore, we were finally unable to resolve precisely how the volume tests compares to the FEC's standard of 163 hours MTBF, though it is clear that the failure rate observed during volume testing was very high.

These calculations provide evidence that the failures observed during the July 20th test are serious. It is hard to escape the conclusion that any system with failure rates this high is not ready for use in an election.

### *Calculation of the Mean Time to Failure*

In our analysis of the Mean Time Between Failures (MTBF) we assume that the failure rate is constant; this is appropriate in situations where there are neither "burn-in" nor "wear-out" effects, as is the case here. We also assume that failures occur independently of one another, i.e. one failure does not increase or decrease the probability of another. From this it follows that the failure process is memoryless, i.e., the probability of failure for any period of time does not depend on the past history of the machine. Mathematically this is equivalent to assuming that that the number of failures in a given time period has a Poisson distribution, and that the length of time between failures has an Exponential distribution.

There were 96 machines tested for 5.33 hours each. This corresponds to 511.68 total test hours. There were 34 failures of various types.

Including all 34 failures, the estimated MTBF is 511.68/34 = 15.05 hours. Given our assumptions, this is also an estimate of the mean of the underlying Exponential distribution. Based on the data available, we can say with 95% confidence that the MTBF is between 10.8 hours to 21.6 hours. The MVTF is 10720/34 = 315 votes.

Considering only the 20 software failures, the estimated MTBF is 511.68/20 = 25.6 hours. The estimated failure rate (mean number of failures per unit time) is 1/25.6. Considering only software failures, the 95% confidence interval for MTBF is 16.6 hours to 41.9 hours.

There were 10,720 votes cast during the volume testing. Considering only the 20 software failures, the estimated MVBF is 10720/20 = 536 votes.

### *Analysis of Printer Jams*

The first category of failures observed were printer jams of various kinds. Most of these were paper jams. Generally speaking, printer jams can be classified into "bottom jams" and "upper jams". Blank paper is fed from a paper supply reel (the "bottom" of the paper path) to the printer, where the VVPAT record is printed. The paper then travels past a window where it can be inspected by the voter, and finally on to the paper take-up reel in the security canister (colloquially, the "top" of the paper feed path). A bottom jam is one that occurs in the part of the paper feed path before the paper enters the printer, e.g., between the supply reel and the printer itself. An upper jam is one that occurs after the printer, e.g., between the printer and the take-up reel.

**Results.** There were 14 printer jams in all. Unfortunately, it is difficult to classify these further: according to the data we have, there was 1 reported upper jam, 3 reported bottom jams, and 10 paper jams whose location was not specified.

Many of the printer jams caused a loss of VVPAT records. For 13 of the 14 jams, we were provided with a final count of the number of VVPAT records on the affected machine as well as a final count of the number of electronic ballot images recorded. (For machine #42, no count of VVPAT records or of electronic ballot images was provided.) For 6 of these 13 cases where counts were available, the number of VVPAT records matched the number of electronic ballot images, and so we assume no VVPAT records were lost. However, for the other 7 jams, some number of VVPAT records were lost. These involved 6 distinct machines; machine numbers #8, #10, #33, #45, #55, and #60 were associated with 1, 5, 4, 2, 1, and 8 lost VVPAT records, respectively. Machine #10 experienced two printer jams. In total there were 21 lost VVPAT records, out of a total of 1535 ballots cast on those particular machines.

In every case where a printer failure occurred, the loss of VVPAT records would be evident upon inspection of the paper trail. In every such case, the paper stopped advancing and the printer overprinted multiple lines of text to the same place on the paper. This condition would be easily noticeable by anyone inspecting the VVPAT records. Further, in such a case the number of lost VVPAT records could be easily bounded by reconciling the number of ballots cast against the polling place rosters.

**Risks.** The loss of VVPAT records would be problematic during any recount of the VVPAT records. If the VVPAT were to govern in the event of any discrepancy between the electronic and paper records, as is apparently required by the newly signed bill SB 370, then lost VVPAT records might constitute lost votes.

### *Analysis of Software Failures*

There were also many software failures observed during the June 20th tests. Frankly, we find the software failures more problematic than the printer jams. The software failures took several forms. For some of the failures, the machine reported a fatal error and was unable to proceed. Other failures left the machine stuck, hung, or frozen in some state and unresponsive to voter input. Generally speaking, there were two broad categories of failures. In the first category, the software failure occurred after a ballot was cast. In the second category, the machine froze immediately after a voter activation card was inserted, and the screen continued displaying the same message (indicating successful casting of a ballot) left over from the previous voter.

In all cases, the voter was provided feedback on whether their ballot had been cast or not, and the VVPAT records were consistent with the electronic records in each case.

**Results.** In all, there were 20 software failures. Two machines experienced 2 software failures in close proximity, which suggests that the existing recovery procedures may not always repair the error condition.

**Risks.** In general, we are concerned that the prevalence of software failures during the June 20th test may indicate software quality problems in the TSx. It is possible that these failures are a sign of a large number of other latent software defects. As far as we know, there has never been another volume test of the TSx that tests the machine under realistic conditions, and generally at best spotty records kept of any failures that may occur during elections, so there is no way to know the extent or magnitude of the software quality problem.

It is possible that votes could be lost or corrupted by software failures such as those detected in the June 20th test. For instance, there were failures where the TSx crashed or hung when attempting to cast a ballot or to remove the voter smartcard, and these could easily have led to the ballot going unrecorded or recorded inaccurately. In the worst case, vote files could be corrupted or truncated when software failures happen. We believe that this issue warrants further investigation before any modified versions of the TSx are certified.

The fundamental barrier to analysis of these software errors is the lack of access to source code for the TSx. With access to this material, it would be possible to identify the cause of each software failure, diagnose the defect in the software, and ascertain the magnitude of the defect. Lacking source code, though, we have no way to perform such an independent evaluation. This is a very unsatisfying position to be in.

We believe these failures constitute one of the strongest arguments for the State of California to take possession of, or otherwise arrange for unfettered access to, the full source code and binary executables for all electronic voting machines. In the absence of access to source code, the State may wish to consider demanding from the vendor a comprehensive itemized accounting of the cause of each software failure, complete with enough technical details that independent technical experts can confirm the vendor's account for each. Even with such an accounting, however, is that there is no way to know whether the defects have been fixed satisfactorily (as opposed to just hidden), or whether they represent symptoms of more serious architectural flaws, without access to the source.

## Recovering from Failures

When failures occur, poll workers need to have some way to recover from, or otherwise respond to, the failure. Of course, different kinds of failures require different recovery strategies. We believe that some procedures are needed to deal with the most common kinds of failures. In this section, we discuss several options for recovery.

There are several challenges with repairing machines in the middle of a running election. First, the amount of training that poll workers receive is limited; if repair is a delicate operation then repairs may not always be performed correctly. Second, there is not always a lot of time to perform detailed operations during a hectic election day. Finally, the biggest fear is that a failed repair may actually make things worse: for instance, certain kinds of software failures might cause the vote file to be corrupted invisibly, and then continuing to vote on that machine not only destroys the chance of fixing this corruption through careful forensic analysis, but it may actually lead to overwriting or destruction of prior and future votes. We discuss each of these risks in detail later. For these reasons, the safest repair is almost always to take a machine out of service and inspect it after the election when there is time to do the job right. Generally speaking, the greater the rate of failures, the greater the need for advance planning on how to recover from them. This leads to two broad policy options for dealing with failures.

**Option 1.** *Ensure that failures are very rare, and adopt simple procedures*. This approach requires that the certification process verify that the failure rate of the machine will be very low. Once this determination has been made, there is little need for sophisticated failure recovery strategies, and one might specify that any machine that encounters a failure is simply taken out of service. No complicated poll worker training is needed; and it is easy to have confidence that a failed machine will not affect future votes.

**Option 2.** *If failures are expected, develop detailed and sophisticated procedures for failure triage, repair, and recovery.* These procedures might specify how to recognize different kinds of printer jams or crashes, which kinds can be repaired and how to repair them, and which kinds cannot be safely repaired. These procedures must ensure to a high level of confidence that the recovery strategy will preserve the integrity of the vote and the VVPAT, will not violate voter privacy, and will reliably bring the machine back to proper working condition. However, this approach does have significant disadvantages. For many of the failures observed in practice, developing safe recovery strategies is delicate and tricky business, so there is a heightened risk that the recovery strategy itself will fail or perhaps even make things worse from time to time. Moreover, this approach places a significant additional training burden on poll workers.

There is a continuum between these two extremes. The decision of what approach to follow will likely need to be made on pragmatic grounds. Our inclination would be to recommend striving to get as close to Option 1 as possible, and wading into the morass of recovery and repair only when necessary, but such decisions will need to make on a caseby- case basis.

### *Recovering from Software Failures*

**Availability.** We note that, if failures are common, taking failed machines out of service may lead to a very substantial shortage of machines. For instance, suppose that our response to all software failures was to take the machine out of service (and in the absence of fairly exhaustive understanding of the code, this is generally the safest response). Then the software failure rate observed during the June 20th volume tests would suggest that a significant number of machines would have experienced at least one failure by

the end of election day, and have been taken out of service. Some precincts— particularly those with only a single machine per precinct—might have been left with no working machines by the end of election day. Voters at those precincts might have to cast provisional ballots but the disruption, especially for disabled voters, could be considerable.

Our calculations suggest that, if machines are taken out of service upon any software failure, then the software failure rate observed on June 20th is untenable. Either the software quality must be dramatically improved, or strategies for recovering from software failures must be developed, or both.

**Reboot and recovery.** Unfortunately, it is not at all clear how to safely recover from arbitrary software failures. The most obvious strategy might be to reboot the voting machine and continue using it. The problem, though, is that rebooting and continuing on is in general quite dangerous in a mission critical situation like an election.

What's wrong with rebooting after an arbitrary error? We just don't know. Without knowing the error and without study of the code, we are flying blind. We cannot even list the things that *could* go wrong, let alone estimate the likelihood of any of them. One problem is that it might lead to the destruction of votes. If the software crashed just as it was writing to electronic storage, it is possible that the vote file system might become corrupted. In this case, rebooting and continuing on could, depending upon the precise error and the file format, lead to greater corruption. In this scenario, rebooting turned a software failure that affected at most one voter into an incident that could potentially affect every vote cast on the machine. This is certainly a worst-case scenario; however, it is not impossible.

Another problem with rebooting is that it might make it much harder for forensic methods to track down the cause of the error and to recover the vote that was being cast (if any) during the failure.

Of course, it is also possible that none of these would happen. There are certainly failure modes where rebooting is reasonable. If the cause of the failure is known in advance, then it can be studied, and in many cases it may be possible for technical experts to confirm that rebooting is a demonstrably safe recovery strategy for this specific cause of failure. For instance, we might be able to verify that some particular bug can only be triggered during initial vote selection, before anything has been indelibly printed or written to permanent storage, and verify that rebooting will return the system to a knownsafe state.

In general, for bugs that have been seen before and have become familiar, it will be possible to investigate them, and recommend safe procedures for recovery. But then again, bugs that are frequent enough to be familiar *really ought to be fixed*, and hence there should be none of them! Hence, we should *presume that all failure and crashes areunexpected*, and for unexpected failures, the only truly safe way of dealing with them is to immediately take the machine out of service.

No matter what recovery strategy is adopted, we recommend that the method for dealing with failures be thought through in advance and documented in detail as part of the procedure manual for that particular voting system. Like any other procedure, recovery procedures affect the integrity of the voting system as a whole. It would be appropriate for evaluation of whether the suggested recovery procedures are fit for purpose to be included within the scope of the certification process.

We suggest that the vendor provide supporting documentation and present evidence that the suggested recovery procedures are safe and adequate. The procedure manual that was submitted as part of the application for certification of the TSx is deficient in this regard, providing no guidance on how to deal with software failures.

It would be helpful if the DRE were to generate an audit log entry any time it is rebooted, and for this audit log entry to contain the number of ballots currently stored in the ballot file and the time at which the reboot was requested. It would also be helpful if the DRE were to try to generate an audit log entry (including the number of ballots currently stored, and information about the error) before crashing, where possible, though we recognize that this may not be feasible for many kinds of software failures.

**Re-voting.** Another vexing problem with recovering from software failures is that there is no clear, general way for a poll worker to determine whether the voter's vote was recorded before the failure. We are very concerned that many of the software failures encountered in the June 20th test violate this crucial requirement. Some of the failures led to a crash after a ballot had been cast; some led to a crash before the voter had begun voting; yet the screens displayed after the crash gave no obvious way to distinguish between these two situations. What can a poll worker do when they are called over after such a failure?

The problem is that in many cases the poll worker has no way to tell whether the voter has actually cast a ballot or not: the machine provides no indication of this essential fact, and often cannot do so. This leaves the poll worker with a difficult conundrum: if a ballot has been cast, then the voter should not be allowed to vote again; but if not, then the voter definitely should be permitted to vote again. Yet in many case these two scenarios are essentially indistinguishable, at least with respect to any evidence the poll worker may have. The poll worker is in a tight spot with little assistance. We studied the suggested procedures for use of the TSx, and we did not find any guidelines that would help a poll worker decide how to resolve this dilemma.

Also, voters who were affected by these failures might become upset, and understandably so, if they were to realize that there is no way to tell whether their vote had actually been recorded.

We do not have a good solution to this problem. It seems the only effective defense is to make the machines as reliable as possible, so that software failures are as rare as possible. When software failures do occur, the most plausible option we can see is have the voter vote again provisionally and decide later whether to count the provisional ballot, but it is not clear that waiting will provide any additional guidance about whether the provisional ballot should be counted or not.

***Recovery from Printer Jams***

**Risks.** Recovering from printer jams generally involves opening up the printer unit and manipulating the paper tape. This is a sensitive operation. It is analogous to opening up a paper ballot box in the middle of an election and inspecting and manipulating the ballots contained therein. If not performed properly, this could endanger vote integrity and privacy.

Fixing paper jams also poses risks to ballot secrecy, as the poll worker performing this step has an opportunity to see the contents of previously printed VVPAT record. A poll worker who is left to fix a paper jam on her own could potentially inspect several VVPAT records (if the jam prevented them from being spooled into the security canister), and because these are recorded in sequential order, the poll worker may be able to connect these with specific voters. Even a well-intentioned poll worker may be unable to avoid noticing the contents of the VVPAT record most recently printed, and this reveals how the last voter voted. As the State Consultant's Supplemental Report regarding the TSx stated, "In this [upper] jam condition, the audit records are visible to the poll worker clearing the jam."

**Mitigations.** One effective way to mitigate these integrity and privacy risks would be to apply two-person controls whenever the printer unit must be opened to clear a jam (or for any other reason). This process could be monitored by at least one other poll worker, who could ensure that the procedure is performed correctly without deliberate peeking. Repairs should not be performed out of sight of observers.

It would also be helpful to have poll workers keep a log any time they clear a paper jam. The log entry would list the cause of the error, the date and time, the machine ID, the identity of the poll workers who performed the repair, and how many VVPAT records were destroyed or otherwise affected, if apparent. Also, it would be helpful for DRE machines to generate an audit log entry whenever the printer unit is opened or closed. The audit log should be designed so it is possible to determine the number of ballots cast before each such event.

The proposed procedures manual for the TSx, as provided in the application for certification, mandates that at no time should the security canister ever be opened (e.g., to clear a paper jam). Instead, it suggests that the canister be replaced if necessary, and the old canister be stored in the election returns bag. We laud this provision; it seems thoughtful and appropriate.

***Generic Strategies for Dealing with Failures***

In general, it may be useful to establish a procedure for poll workers to keep records on all failures or anomalous events of any kind (including both printer jams and software failures). The poll worker might record the time, what went wrong, and what remediation was used. This would make it possible to perform analysis of these records after the election and to measure the reliability of voting machines as they are used in the field. This kind of quantitative-driven reliability assessment would enable election officials to focus in on the most common failures, to compare the reliability of different systems, and to plan for future elections. Such an effort might lead to long-term improvements in overall system reliability.

It may make sense to mandate that a manual audit of the VVPAT records be performed for any machine that experiences a crash or software failure of any kind, and any discrepancies be investigated thoroughly and reconciled. These audits would follow the same process as the 1% manual recount audit, but audits of failed machines should be in addition to the precincts selected for the 1% audit, not a replacement for part of the 1% audit. The reason is that the effects of software failures are generally unpredictable, and the vote file stored on a machine that has experienced a software failure is inherently somewhat suspect.
In general, we recommend that procedures be developed regarding how to handle failures during election day. A good guiding principle would be: If a failure occurs that is not covered by the procedures, a call to a county official should be placed for guidance, or—failing that—the machine should be taken out of service. These procedures might also specify other conditions when a machine is automatically taken out of service (e.g., if it experiences multiple failures).

**Refining the Volume Testing Methodology**

We believe the June 20th test provides an excellent framework for volume testing and for reliability testing in general, and we recommend that this framework be adopted for future reliability tests, with several refinements described below.

**Expanded data collection.** First, collecting a little bit of additional data regarding failures would have made it possible to perform more powerful data analysis. In addition to the fields collected on the June 20th test, we recommend collecting several additional fields about each incident:
1. the wallclock time at which each failure occurred;
2. the number of votes cast/stored up until this point (this might be obtained, e.g., by reading the protective counter);
3. the identity of the tester whose testing caused the failure;
4. answers to several standard questions, to be asked of the tester after each failure:

- At what stage in this process did the failure manifest itself?
- Did you see a printed VVPAT record appear?
- Had you hit the “cast” button?
- Did it print “Verified” on the VVPAT record?
- Did the screen provide confirmation that the ballot was cast successfully?

5. a photograph of the screen when the error occurred and,
6. a more detailed description of each failure, if possible (e.g., in the case of paper jams, where did the jam occur, and how many VVPAT records were affected?).

Second, it would be very helpful if the fields listed on the June 20th data reports were filled out more consistently. Many fields were missing or not filled out. For instance, the “time” field was often left blank. Also, some fields (e.g., the cause/type of error) were filled out in more detail by some recorders than by others.

In addition to information collected when a failure is encountered, we suggest collecting some additional information from each machine after the end of the testing period. For each machine, this would involve collecting:

1. the number of electronic ballot images stored in the vote file;
2. the number of VVPAT records marked as “Verified” that are stored in the canister;
3. the number of VVPAT records marked “Spoiled”;
4. the number of VVPAT records that are marked as neither “Verified” nor “Spoiled”, if any;
5. a copy of the audit log file, containing all audit log entries; and,
6. a copy of the vote file, containing all electronic ballot images.

Collecting this data is somewhat labor-intensive, but it enables more detailed data analysis.

**Parallel testing.** It would also be desirable to further improve the reliability testing by recording more in-depth information about a limited subset of machines. In particular, we suggest that one might apply the same procedures used during parallel testing to the selected subset, such as recording with a video camera everything that is done to the selected machines during testing. The potential advantage of this is that if any of the selected machines experience a failure, it will be possible to review the video tape to see exactly how the failure occurred and to assess more precisely its effect on the electronic ballot image and the VVPAT record. For instance, one could review the ballot selections made by the tester, then cross-check them against what is stored in the vote file, in the VVPAT records, and in the audit log.