Automating Voting Terminal Event Log Analysis

Tigran Antonyan Seda Davtyan Sotirios Kentros Aggelos Kiayias
Laurent Michel Nicolas Nicolaou Alexander Russell Alexander Shvartsman

{tigran,seda,skentros,nicolas}@engr.uconn.edu
{aggelos,ldm,acr,aas}@cse.uconn.edu

Voting Technology Research Center
and Computer Science and Engineering Department
University of Connecticut

Abstract

In the interest of auditing election procedures, certain electronic voting technologies provide monitoring capabilities that record select actions undertaken by election officials before, during, and after an election process, as well as the conditions present in an electronic voting terminal as the result of its interactions with its environment. In this paper we report on an automated auditing process for detecting procedural irregularities for elections employing the AccuVote Optical Scan (AV-OS) terminal (manufactured by Premier Election Systems).

Our auditing process is derived from an abstract finite state model of the AV-OS; this determines, in particular, a correspondence between state transitions and logged events that separates expected and "irregular" histories. Automating the detection of these irregular histories has permitted us to provide detailed election procedure audits for full-scale Connecticut elections. We conclude the article with a discussion of the result of the event log analysis performed within the post-election audit of the November 2008 elections in Connecticut.

Additionally, we identify a defect and some deficiencies in the AV-OS event logging subsystem that can interfere with the event log transcript making it vulnerable to manipulation and discuss the effects of these deficiencies.

This research is funded by the Office of the Secretary of the State of Connecticut.

1 Introduction

Post-election audits are essential to ensure voter confidence in the outcome of elections. Many states mandate audits and official recounts when the number of votes cast for leading candidates are within a close proximity (e.g., within 0.5% in Connecticut), or when the election results are disputed. In addition, it is now considered important to conduct random audits regardless of the perceived outcome [12], and some states mandate post-election audits for all general elections. The broad introduction of new, and in some cases untried, electronic voting equipment prompted by the Help America Vote Act (HAVA) [5] underscores the need for independent examinations of the election outcomes. Recent research provides strong evidence that the security and dependability concerns associated with electronic voting technology are very real (e.g., [11, 10, 13, 15, 6, 7, 14, 4, 8, 9]). Consequently, nationwide discussions involving various public officials and diverse civic groups have been focusing on the need to enable and to perform meaningful election audits, and in particular, audits of electronic voting technologies used in the elections. It is noteworthy that while concerns about the electronic voting terminals persist, electronic voting also enables the capture of valuable data pertaining to the conduct of an election in addition to reporting the final tally. For example, electronic voting terminals are normally equipped with the capability to record an *event log* that contains timestamped entries corresponding to significant actions performed by the terminals and their users. The ability to analyze event logs offers a valuable capability within any approach to election audits.

This paper presents our approach to and work on automating the analysis of the event logs of electronic voting terminals. The University of Connecticut VoTeR Center (Voting Technology Research Center) is an independent entity, and we perform our work within an established relationship with the Connecticut SOTS Office, that chartered us to examine and evaluate electronic voting technology, to assist the State in defining safe use procedures for the electronic voting terminals, and to perform audits of technology before and after each election. Our previous work related to audits of technology is presented in [2].

We have implemented a tool that allows for the event logs collected in the wake of an election to be methodically analyzed, with the goal of detecting any deviations

from the prescribed and recommended use of the voting terminals in the election. On the request of Office of the Secretary of the State (SOTS) of Connecticut, we used this tool after the November 2008 election to perform a detailed analysis of the voting terminal audit logs representing over 30% of all precincts in the State. We present a summary of our findings in this report, with the conclusion that detailed event log analysis should be included as an essential part of any audit process.

We believe the approach to event log analysis presented in this paper is very general. Naturally many electronic voting technologies incorporate monitoring capabilities that enable retrieval and examination of the actions performed on and by the system during an election. These actions are normally recorded in preallocated memory space that we refer to as the system's *event log*. We present our approach in the context of a specific voting terminal, Premier's Accu-Vote Optical Scan (AV-OS) terminal (firmware version 1.96.9), that is used in Connecticut elections since 2006. Specifically, we present a part of the post election audit process that deals with the analysis and examination of the event logs collected from the AV-OS machines.

The event log of an AV-OS voting terminal is stored in a removable memory card [2]. While the voting terminal hardware is identical in all Connecticut precincts deploying AV-OS systems, the contents of the memory cards differ depending on the specific elections in individual precincts. The event log within an AV-OS terminal contains a history of actions since the memory card was formatted and programmed for the specific precinct.

We note that, for the purposes of this work, we treated the AV-OS terminal as a "black box". The results derived and presented here are obtained through extensive testing procedures using the functions provided by the machine, and through observations of how the terminals' use and behavior is recorded in its event logs.

This paper emphasizes the importance of including the event log analysis as a part of a post election audit of technology. A careful examination of events and corresponding timestamps reveals information about the procedures followed during an election process, including the information suggesting improper conduct of an election or malfunction of terminals.

Related Work: The AV-OS voting terminal has drawn the attention of a number of research papers that demonstrated security vulnerabilities in the system [6, 8, 9, 3, 2]. The report of Hursti's [6] revealed that the AV-OS memory card lacks cryptographic integrity checks, thus it can be maliciously manipulated and exploited with the help of specialized hardware. These findings led several jurisdictions employing the system to use tamper-evident seals in securing the memory card in the terminal. The Connecticut SOTS commissioned a follow-up study to confirm Hursti's findings and identify other vulnerabilities. Our study [8, 9] demonstrated that an additional attack can be successfully launched against the AV-OS even if the memory card is sealed in. Here the attack exploits a communication port on the machine and by commandeering its card writing capability, it transparently modifies the contents of the memory card. The same attack also exploits vulnerabilities of the machine's proprietary language, called AccuBasic, used for reporting election results.

Although previous attacks assumed that the firmware of the AV-OS terminal is a trusted component of the system, subsequent reports [2, 3] proved this assumption to be incorrect. In particular the two reports show that someone with physical access to an AV-OS terminal may manipulate the firmware code in a way that will be undetectable during election day testing. While [3] demonstrates both benign and malicious manipulations of the firmware, [2] underscores and presents procedures for pre-election and post-election testing to minimize and/or eliminate any vulnerabilities that potentially may alter the election results. Our approach leads the way towards ensuring integrity of the electoral process, over unreliable and untrusted hardware. The results of this work and the techniques developed by us were used in the audits of technology in all elections in Connecticut starting in 2006. In this paper we present a new step that enhances election audit procedures presented in [2] with a detailed examination and analysis of the AV-OS event logs.

Contributions: Our goal is to introduce event log analysis procedures that facilitate the efficient and accurate analysis of the event logs of the AV-OS terminals or any electronic voting terminal that provides the event logging capability. Our development involves the design and implementation of a software tool that automates the event log analysis. For this purpose we define and describe a rigorous computational model, embraced by any electoral process that deploys AV-OS voting terminals. We map our model to a state diagram and from here we introduce action chain rules that must be satisfied by any consistent election process. Based on our extensive testing and the analysis of event logs we also discover and report deficiencies in the actual logging process of the AV-OS terminals. In more detail our contributions are as follows.

- We develop a model that describes the action chains and specifies the proper stages in an election process. Based on this election model we develop a state diagram that represents the flow of actions in the election process and that identifies acceptable

and unacceptable action sequences that should be the target of the event log analysis procedure.

- We present the types of actions that the event log is able to record, then we show the mapping of the actions to the state diagram and we introduce event rules that should be followed in an election process.
- The specification of event rules facilitated the development of an event log analysis tool that reports any inconsistencies or deviations from the prescribed rules based on the event log data. The tool automates the analysis of a large number of event logs, thus expediting this part of the election audit.
- Based on our experimentation with the AV-OS voting terminal, we report on defects and deficiencies in the action recording procedure used by the AV-OS. In particular we discover a log overflow vulnerability as well as inadequacies in the date and action recording procedures. The defect in the event log overflow handling may affect the event ordering on the log printout and may lead to malicious event log manipulations. The inadequate date recording may hide the actual execution time of the events. Lastly, inadequate action logging may result in the absence of reporting of certain events.
- Finally, we present the results and our observations based on the application of our automated event log analysis tool in the actual post-election audit conducted after the November 2008 elections in Connecticut.

We believe that audits of election technology, and in particular audits of truthful event logs of electronic voting terminals, are crucial in providing timely information and maintaining the integrity of the electoral process.

We note that our evaluation and tool development was performed without any access to internal vendor documentation, source code, or assistance of any kind.

2 The Election Process in Connecticut

Figure 1 illustrates the election process flow when using an AV-OS terminal. This section describes the election flow in more detail. We believe this model to be fairly general, however it incorporates an election audit model that may be specific to the State of Connecticut (CT).

Before Election Day: Election preparations in CT begin at least 30 days prior to the election day. The memory cards of AV-OS terminals are programmed for each precinct. The voting terminals also undergo routine maintenance and testing to help prevent failures during the election. The memory cards are programmed by a service company contracted by the State. Four programmed cards then are securely transported to each polling location. When the cards arrive, election officials conduct pre-election tests on all the voting terminals with all the cards. Then the officials randomly select two cards. These cards will be used in the primary and the back-up terminals. The two selected cards are sealed in their respective voting terminals and are set for election.

On Election Day: The summary of the activities that take place on the election day is as follows.

- ***Before The Polls Open:*** On the morning of the election day, before the polls open, the poll workers and/or registrars of voters need to verify the seals on each AV-OS terminal, and verify that the machines are properly initialized. This includes making sure all candidate counters are set to zero, by printing a Zero Totals Report.
- ***While The Polls Are Open:*** Each eligible voter is entitled to a single ballot that s/he receives once they are verified against the voter registration database. Once the voter fills the ballot s/he proceeds to feed the ballot to the optical scanner of AV-OS.
- ***After The Polls Close:*** The election officials print the totals report directly from the AV-OS terminal (the event log can also be printed at this point). The printed tape is then delivered to the central tabulation process where the totals are computed and reported to the Secretary of the State Office for certification. Note that in CT, the central tabulation tool available from the vendor of AV-OS (Premier's GEMS) is *not* used. In jurisdictions that use central tabulation, the results either are uploaded from AV-OS terminals to a central server where the tabulation is performed, or the memory cards with the results are transfered to the tabulation center, where an AV-OS terminal is used to upload the results to the tabulation server. Depending on the state, the central tabulation is usually done at the municipal or county level.

Audits: Three independent audits are performed in conjunction with each election (in Connecticut).

- The *Pre-Election Technical Audit* involves the examination of one memory card, chosen randomly from the four cards supplied to each precinct. The audit covers a large subset of precincts. Over the last three years pre-election audits involved anywhere from 25% to 100% of the precincts.
- The *Hand Count Audit* is a post-election audit that consists of complete manual counting of a subset of

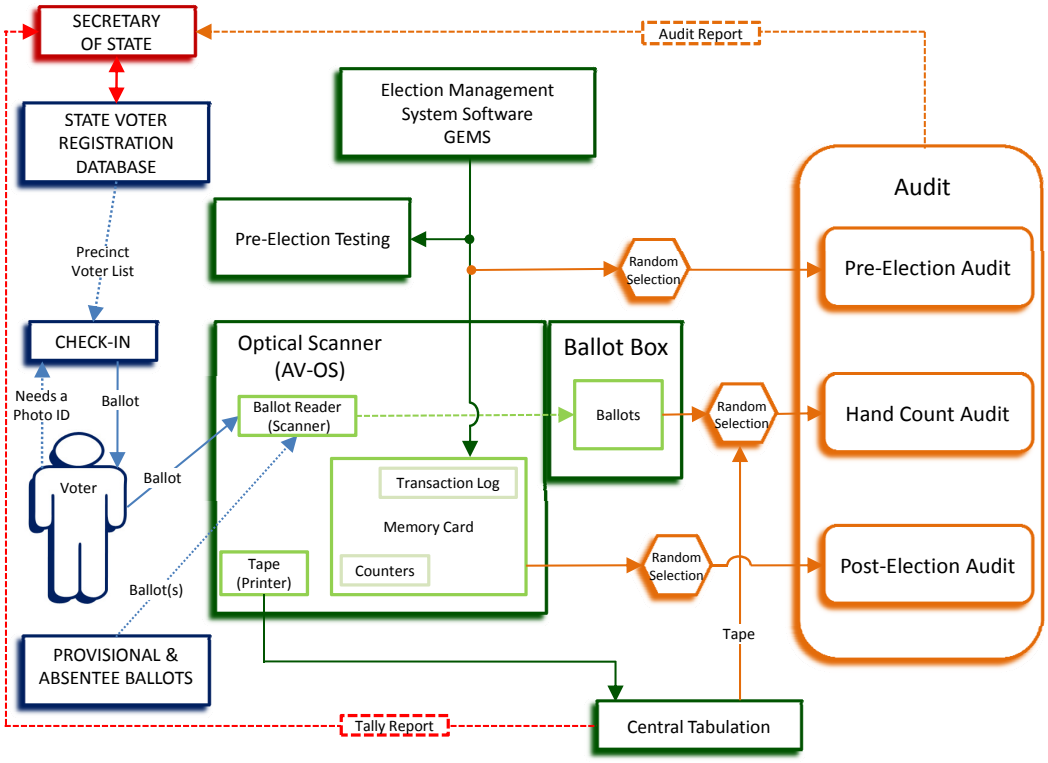

Figure 1: Election Process in Connecticut using AV-OS

races in 10% of precincts randomly selected after each election.

- The *Post-Election Technical Audit* involves the examination of the memory cards used in the election, typically covering up to 30% of the precincts.

The memory card audits include examining the cards for proper programming and absence of extraneous or unexpected executable code [2]. Our current work deals with a detailed analysis of the event logs collected from the voting terminals used in the election.

3 Event Log Analysis

This section presents our modeling of the voting terminal behavior that includes the actions recorded in the event log and our approach to automating the analysis of the logs. We implemented a rule-based tool that inputs an event log generated by a voting terminal and classifies it as either normal, i.e., containing a sequence of events expected within a normal course of an election, or irregular, i.e., containing unexpected events, unusual sequences of events, and unanticipated timing of events. The irregular logs can then be further analyzed manually. Our automated event log analysis tool was used to perform a broad audit of the event logs collected from over 30% of the precincts following the November 2008 election in Connecticut. We present the highlights of the audit in Section 5.

Here we start by presenting the action types recorded by the AV-OS voting terminal in the event log, the states that a terminal may go through during the election process, and the actual traces produced in the event log. The entire system is modeled as a finite state machine that determines the language of valid log sequences. The analysis tool augments the basic finite state machine with time predicates on the transitions to capture the time sensitivity aspect and to model when transitions are enabled or disabled. This is the basis on which the log analysis tool is built. The section closes with a brief description of the actual tool.

3.1 Actions Recorded in the Event Log

Table 1 presents the action types recorded by AV-OS in the event log along with a brief description. The event log has *action* entries and *date* entries. Most action entries are solely characterized by a name and a time of occurrence (no date). Some action entries, i.e., INITIALIZED and SESSION START carry an additional date stored in the adjacent date entry.

Action Name	Action Description
AUDIT REPORT	Appears when an Audit Report is printed.
BAL COUNT END	After the ender card is inserted in an election, this action appears.
BAL COUNT START	Appears when the first ballot is cast in an election.
BAL TEST START	Records the beginning of a test election.
CLEAR COUNTERS	Appears when the counters are set to zero.
COUNT RESTARTED	Appears if the machine is reset during an election, after at least one ballot is cast.
DOWNLOAD END	Recorded during the programing of the card using GEMS, when the download of data is ended.
DOWNLOAD START	Recorded during the programing of the card using GEMS, when the download of data is started.
DUPLICATE CARD	Appears when a card duplication takes place. Present in both the master card and the copy.
ENDER CARD	Records when an ender card is inserted, signifying the end of an election.
INITIALIZED	The 1st action that appears in the Event Log. Date action that appears when one programs the card.
MEM CARD RESET	A memory card reset returns a card in 'not set' status, if it was set for election.
OVERRIDE	Records an override by a poll worker. Used for the insertion of overvoted ballots in CT.
POWER FAIL	If the machine is unplugged or a power failure occurs, this action is recorded.
PREP FOR ELECT	Recorded when the card is set for election.
SESSION START	Date action. Appears every time you reset the machine.
TOTALS REPORT	Appears when a Totals Report is printed.
UNVOTED BAL TST	Appears when an unvoted ballot test is performed.
UPLOAD END	When an upload is completed, this action is recorded.
UPLOAD ERROR	Appears when an upload error is detected.
UPLOAD STARTED	Marks the beginning of an upload.
VOTED BAL TEST	Appears when an voted ballot test is performed.
ZERO TOT REPORT	Appears when a Zero Totals Report is printed.

Table 1: Event log action types

3.2 AV-OS State Diagram

For reasons of security, memory cards are sealed in the voting terminal prior to pre-election testing and for the duration of the election. Thus we model the voting terminal with the inserted cards as a single system from the time the card is sealed in. Figure 2 illustrates the states the system goes through, from the time a card is programmed, until the election is closed and the results are printed.

Remark. The states Blank Card, Election Loaded, Set For election, Print Totals Report, and Election Closed are the states the machine returns to when it is restarted. These states correspond to the first five states of Figure 1 in [14]. Our diagram does not include the last two states presented in the same work, since in Connecticut the results are not uploaded and thus the machines should never enter those states.

Next we provide a detailed description of the transition function. For each state, we indicate what changes take place as a result of a user action and which events are placed in the event log.

3.2.1 States, Actions and Event Log

Each paragraph of this section describes a state and the transitions that can be followed as a result of user actions. Each paragraph is associated with a figure. In the figure, states are represented with ovals and actions with boxes. Arrow heads indicate the flow from the originating state, through the action box and to the resulting state. Each action box is annotated with a tuple $\langle a \| b \| c \rangle$ where a is the user action, b represents the ensuing sequence of machine actions, and c captures the sequence of logged events.

Blank State. The transitions of the blank state are shown in Figure 3. This state represents a memory card that is formatted but contains no election information. One needs a blank card in order to program it using GEMS, after which the card will contain valid election data. This programming also inserts a sequence of events in the (initially empty) event log: INITIALIZED, DOWNLOAD START, CLEAR COUNTERS, DOWNLOAD END.

Loaded Election State. Figure 4 describes the Election Loaded state. Cards should be in this state when they first arrive at the precincts. At this state test elections can be performed or the card can be cleared in order to be reprogrammed. According to Connecticut election procedures pre-election testing is performed on all cards at the precincts. The entity programming the cards will also perform pre-election testing before it ships the

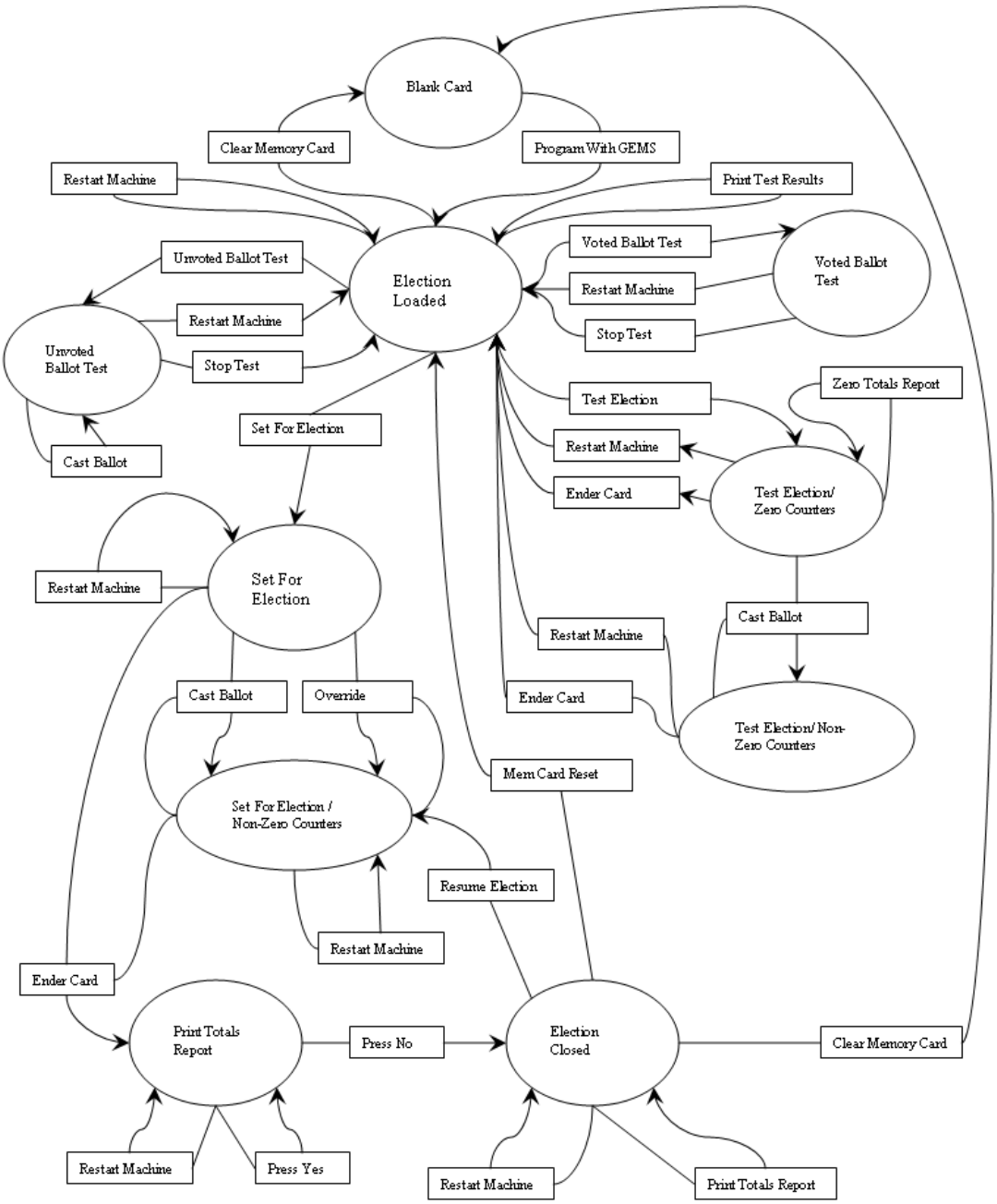

Figure 2: AV-OS State Diagram

cards. From this state, a user can perform the following actions: unvoted ballot test, voted ballot test, test election, clear the memory card, print test results or reset the machine. Most of these actions result in the insertion of a corresponding event in the event log. The two notable exceptions are the voted and unvoted ballot tests.

Voted Ballot Test. Figure 5 illustrates the actions that can be initiated during the voted ballot test. The voted ballot test only accepts a ballot with all its bubbles filled in. If the ballot has some empty bubbles, the voting terminal will report which bubbles were not filled. In all the test states, if one resets the terminal, the card returns to the loaded election state. For the VOTED BALLOT TEST, the same result can be achieved by stopping the test or pressing the no button. Note that the first time a ballot is tested, the voted ballot test event is recorded in the event log.

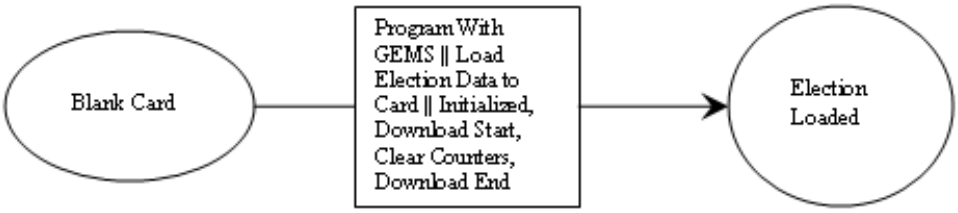

Figure 3: Blank State

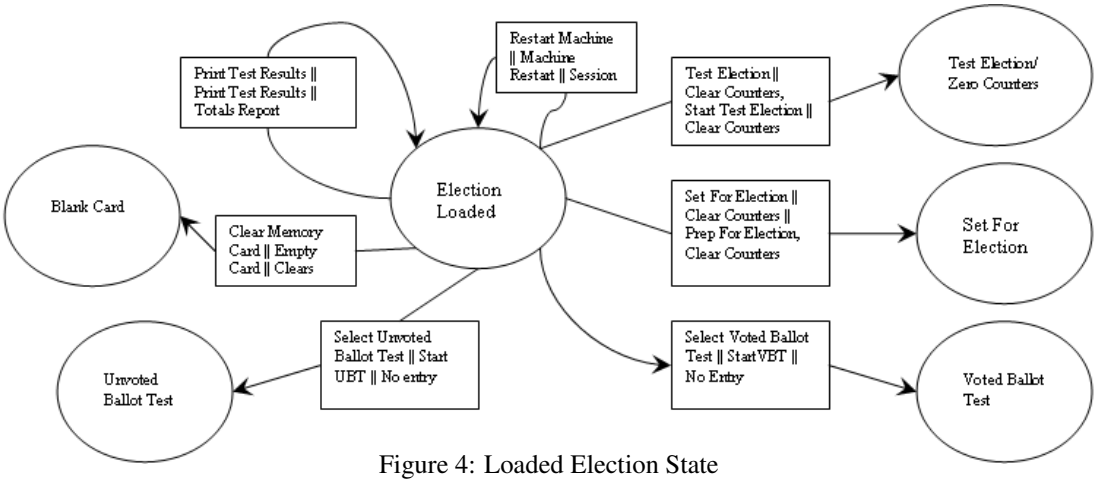

Figure 4: Loaded Election State

Unvoted Ballot Test. Figure 6 illustrates the actions that can be initiated during the unvoted ballot test. The unvoted ballot test only accepts a ballot with all its bubbles empty. If the ballot has some non-empty bubbles, the terminal will report which bubbles were filled. Once again, if the machine is reset, the card returns to the loaded election state. The same will happen if the test is stopped via the no button. The first time a ballot is tested, the unvoted ballot test event is recorded in the event log.

Test Election with Zero Counters. Figure 7 describes the zero counters state and its transitions. We distinguish the test election with zero counters and the test election with non-zero counters, as there is a small difference in the actions that can be performed between them, and some actions lead to the insertion of different sequences in the event log. As before, resetting the voting terminal returns the card to the loaded election state. The same happens if one inserts the ender card. Note that the events logged in the two cases are completely different. Casting a ballot in this state will cause a transition to the test election with non-zero counters state. It also results in the insertion of a ballot test start event in the event log. Finally in this state one can print a zero totals report.

Test Election with Non-Zero Counters. Figure 8 describes the test election with non-zero counters state. Again, resetting the voting terminal returns the card in the loaded election state. The same happens if one inserts the ender card. Note that the events logged in the first and second case are different.

Set for Election with Zero Counters. Figure 9 describes the 'set for election with zero counters' state. This state can be reached from the 'election loaded' state by using the set for election action. A card should be in that state the morning of the elections, before any ballots are cast. A reset of the terminal at this point will not return it to the 'election loaded' state. Instead, the terminal will remain in the current state and print a zero totals report again. Casting any ballot, either with an override or normally, moves the card to the 'set for election with non-zero counters' state and adds the BALLOT COUNT START event in the event log. An ender card ends the election, causing the insertion of the following sequence of events: ENDER CARD, BALLOT COUNT START, BALLOT COUNT END. At this point the voting terminal will start printing the totals report and the card will enter the 'print totals report' state.

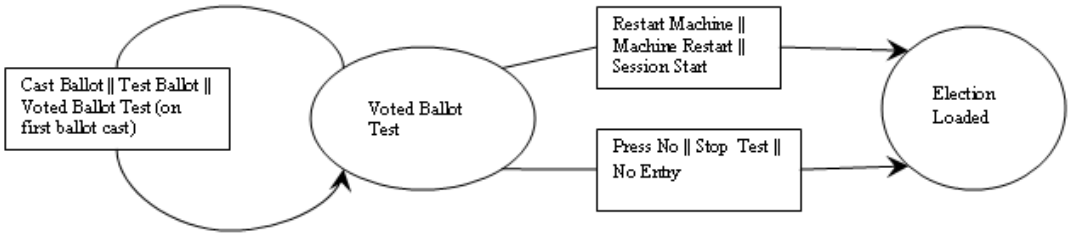

Figure 5: Voted Ballot Test

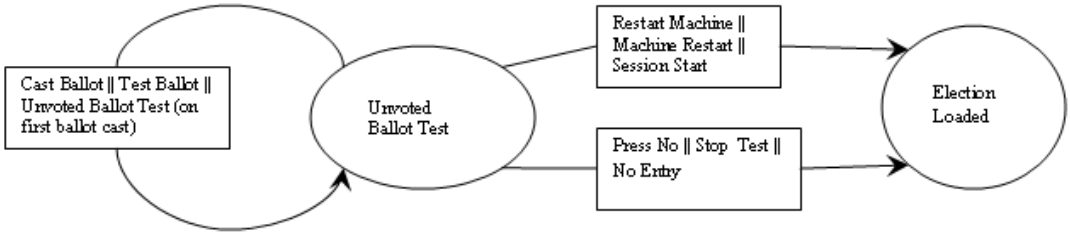

Figure 6: Voted Ballot Test

Set for Election with Non-Zero Counters. Figure 10 details the 'set for election with non-zero counters' state. The card remains in this state as long as the election is ongoing. If the terminal is restarted, the card remains in the same state and this sequence of two events is inserted in the log: SESSION START, COUNT RESTARTED. Voting terminals should not be restarted during the elections according to Connecticut election procedures. If election officials override the machine to cast a ballot, an OVERRIDE event is recorded. Since the overrides depend only on the voters, in a large precinct the event log could contain many OVERRIDE events. An ender card will end the election, causing the insertion of the following sequence of events: ENDER CARD, BALLOT COUNT END. At this point the machine prints the totals report and the card enters the 'print totals report' state.

Print Totals Report. Figure 11 describes the transitions of the 'print totals report' state. If one restarts the terminal, the card will remain in the same state, and it will keep on asking whether the operator wants to print a totals report or not. If one says yes, a report is printed and the terminal returns to the same dialog. If one declines the dialog and only in such case, the terminal enters the 'election closed' state. If at least one report was printed since the terminal was last restarted, a TOTALS REPORT event will be logged in the event log. Note that this makes it possible to have a report printed without an event TOTALS REPORT entered in the event log. This is a defect in the design of AV-OS and is further analyzed in section 4.2.

Election Closed. Figure 12 describes the 'election closed' state. From the 'election closed' state one can resume the last election that ran on the card. Such an action results in the following sequence of events being recorded: PREP FOR ELECTION, COUNT RESTARTED and BALLOT COUNT START. The latter appears on the first ballot cast on the new election. While on the 'election closed' state, one can print more totals reports. Also one could perform a memory card reset action and return the card to the 'election loaded' state. Finally one can clear the card in order to produce a blank card for reprogramming.

3.3 Time Sensitive Rules

The election procedure described in Section 2 is time sensitive. Hence for the purpose of event log auditing, it is important to create time sensitive rules. The event log can be used as a forensic trace with the necessary time information in order to determine which actions were performed at each step of the election process. We highlight below the main time intervals characterizing the election process in Connecticut.

Card Programming and Pre-Election Testing. In Connecticut, the programming of the cards is taken care of by an external entity based on the election data provided by the State. The programming usually starts three weeks before the elections and is completed in a week for almost all precincts. Moreover it is expected that pre-election testing is performed on all the cards prior to their

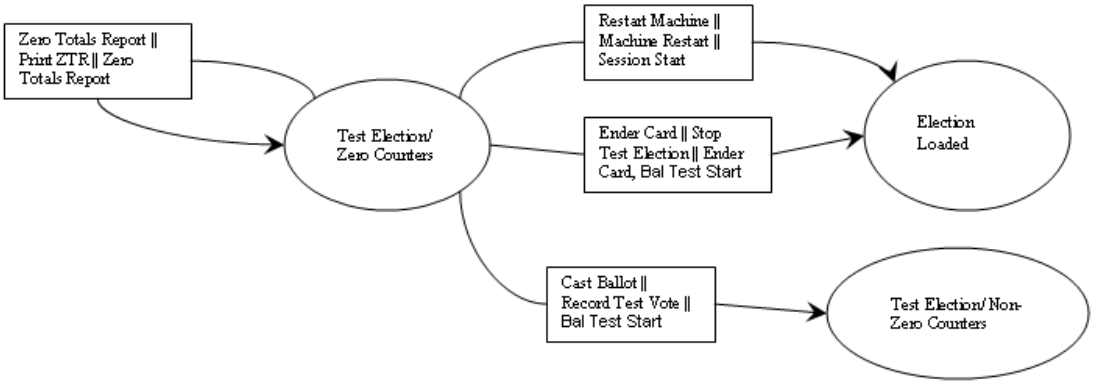

Figure 7: Test Election with Zero Counters

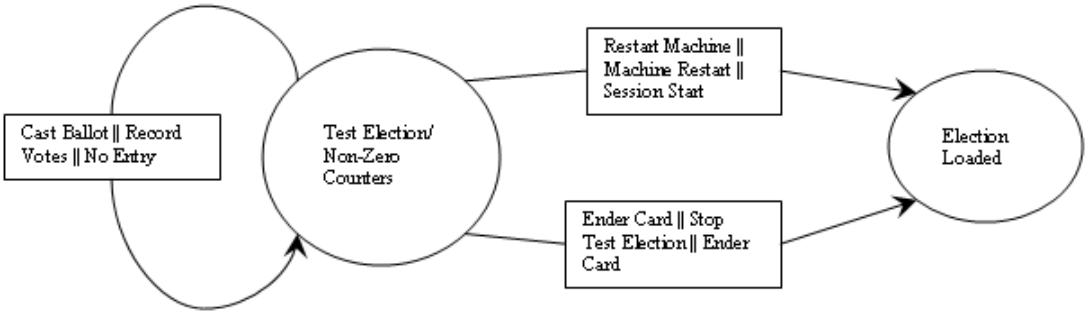

Figure 8: Test Election with Non-Zero Counters

shipment to the precincts. All the events related to the initialization of a card along with some pre-election testing events should occur within that time frame.

Pre-Election Testing and Setting for Election in the Precincts. Cards arrive in the precincts any time one week to two weeks before the elections. The election officials perform pre-election testing on all the cards that arrive. They pick two cards (randomly), seal them in the two AV-OS machines that are available in each precinct, and set them for election. At this time interval we expect a number of events related with pre-election testing and a single PREP FOR ELECTION event. After being set for election, one would expect the cards to not contain any events until the day of the election.

Election Day: The election day imposes strict time limitations on when and what actions can be performed. According to the election process we expect to see the following: from 5:00 to 6:00 AM election officials should switch on the AV-OS terminal to be used in the election and produce a zero totals report. This means that we expect to see a session start (SESSION START) event about an hour before the polls open, followed by a zero totals report (ZERO TOT REPORT) event. We expect to see the ballot count start (BAL COUNT START) event after the time the polls open. This can be followed by a sequence of override (OVERRIDE) events. Finally, by the time the polls close, we expect to see the ender card (ENDER CARD) event, followed by a ballot count end (BAL COUNT END) and a totals report (TOTALS REPORT) events. Thus, on the election day we expect the following sequence of timed events:

1. Session Start, Zero Totals Report (before the polls open).
2. Ballot Count Start (after the polls open).
3. Any number of override events (while the polls are open).
4. Ender Card, Ballot Count End, Totals Report (when the polls close).

3.4 Automating the Event Log Analysis

To automate the event log analysis, we created a tool that takes, as input, both the event log and a set of timed rules derived from the finite state machine presented above. The tool parses the event log string according to these rules to determine if the log represents an "expected" sequence of timed events.

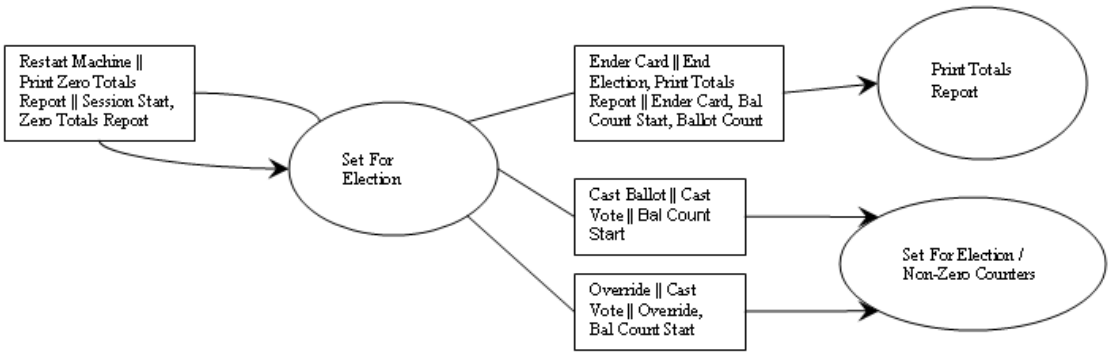

Figure 9: Set for Election with Zero Counters

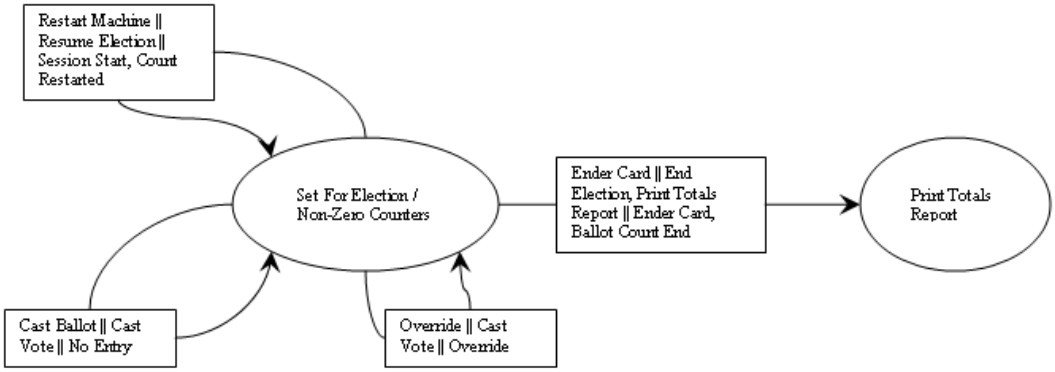

Figure 10: Set for Election with Non-Zero Counters

Recall that the rules define a time-annotated state machine that specifies the language of expected event sequences. Certain transitions are enabled or disabled as a function of the timestamps. For example, test elections can take place in the dates before the election, but not during the election or after the election. So we use the timed rules to bind the normally-prescribed election process and the AV-OS state diagram. If the event log deviates from what is considered expected, it is flagged as "irregular" and we further investigate its inconsistencies.

Table 2 shows three examples containing excerpt from election day event logs. Excerpt A shows an expected sequence. Excerpt B is an irregular sequence that was flagged because it has a voting terminal reset during the election, at 11:20 am. Excerpt C is another irregular sequence that was flagged both because it had a terminal reset at 13:17, but also because the election was never ended. From the inspection of the event log, we can see that the voting terminal had power issues (consequently the poll officials had to switch to the back-up terminal).

4 AV-OS Event Log Defects/Deficiencies

This section describes one defect and two deficiencies of the event logging feature of the AV-OS voting terminal. Our findings are based on repeated and intensive tests that examined numerous event log reports. The defect appears to be related to an undetected overflow condition where the logging module of the AV-OS partially overwrites some entries of the log resulting in erroneous log printouts. The two deficiencies stem from design weaknesses of the logging module that fails to record certain events and prevent any forensic analysis of the log to unambiguously determine which sequence of events lead to the observable content of the log. Our findings underscore the need for a complete analysis of the event logging module of the AV-OS.

4.1 Printing an Overflowed Event Log

The event log printed by the AV-OS consists of a sequence of entries. Of these there are two types: a) *action* entries and b) *date* entries. An action entry is a triple (s, n, t) where s is the sequence number (starting at 1),

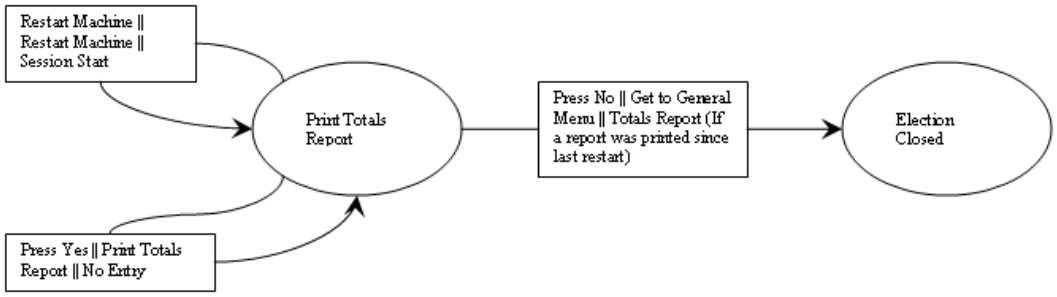

Figure 11: Print Totals Report

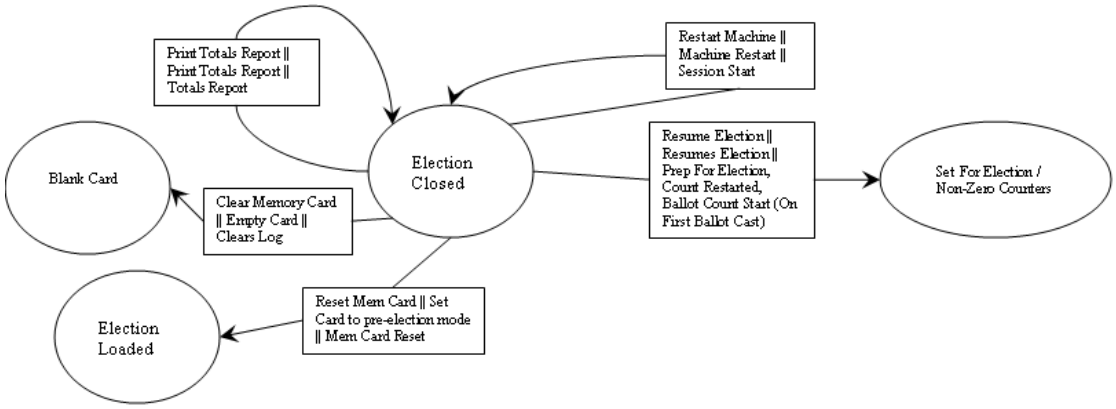

Figure 12: Election Closed state

A. Expected Events	B. Flagged Events	C. Flagged Events
06:45 SESSION START	06:45 SESSION START	05:17 SESSION START
11-04-08	11-04-08	11-04-08
06:51 ZERO TOT REPORT	06:49 ZERO TOT REPORT	05:25 ZERO TOT REPORT
07:05 BAL COUNT START	07:02 BAL COUNT START	06:01 BAL COUNT START
21:07 ENDER CARD	11:20 SESSION START	08:10 POWER FAIL
21:07 BAL COUNT END	11-04-08	10:47 POWER FAIL
21:11 TOTALS REPORT	11:20 COUNT RESTARTED	13:17 SESSION START
	11:21 BAL COUNT START	11-04-08
	21:03 ENDER CARD	13:17 COUNT RESTARTED
	21:03 BAL COUNT END	13:20 BAL COUNT START
	21:13 TOTALS REPORT	14:44 POWER FAIL

Table 2: Examples of expected (A) and flagged (B and C) event logs

n is the name of the action, and t is the time at which it occurred. The action entries SESSION START and INITIALIZED are always followed by a date entry. That date entry records the date of the preceding action in the format DD-MM-YY.

During an election, the internal log will often consist of several dozen entries recording a trace of the interactions of the terminal with its end-users. When the voting terminal is audited, one can request a printout of the internal log.

A natural question regarding the robustness of the logging module is how it behaves when subject to a large number of (perfectly legal) interactions, resulting in an unusually lengthy log. Such an experiment reveals that the internal log is a fixed-size circular buffer of 512 records. When more than 512 interactions occur during an election process, the newest entries overwrite the oldest ones. Despite this, one might expect the system to faithfully report the portion of history reflected in the log. That is, one might expect that with, say, 522 entries, the newest 10 of which have overwritten the oldest 10,

the printed log would indicate the following sequence:

$$(11, n_{11}, t_{11}), (12, n_{12}, t_{12}), \ldots, (512, n_{512}, t_{512}), \\ (513, n_{513}, t_{513}), \ldots, (522, n_{522}, t_{522})$$

However, in the AV-OS, this is not the case. The printout instead shows:

$$(1, n_{11}, t_{11}), (2, n_{12}, t_{12}), \ldots, (502, n_{512}, t_{512}), \ldots, \\ (512, n_{522}, t_{522}), (513, n_{11}, t_{11}), \ldots, (522, n_{22}, t_{22})$$

In essence, it appears that the printing module does not notice the wrap-around and duplicates the entries from the prefix at the end. The immediate side-effect is that the paper printout reports action events with erroneous sequence numbers, out of order and duplicates some of them.

Perhaps even more surprisingly, we observed that if sufficiently many (i.e., 512) *pure* action entries are created, the printed log becomes valid again. Namely, the entries are printed in their true order and the artificial duplication effect disappears.

This observation suggests that, indeed, the logging module detects overflows and behaves like a circular buffer once it sees sufficiently many action entries. It further suggests that action and date entries are recorded separately in the internal log and each consume one slot in the circular buffer. This is not taken into account for overflow detection. For instance, consider the situation where the log has 20 date entries and 500 action entries, for a total of 520 entries. The size of the buffer is 512, which results in an overflow of 8 entries. The newest 8 entries overwrite the 8 oldest entries of the log. However the overflow remains undetected until the number of action entries exceeds 512, and we would thus observe the irregular behavior described above in the event log printout. So if, for example, 1 date entry and 7 action entries are overwritten the printed event log will show:

$$(1, n_8, t_8), (2, n_9, t_9), \ldots, (493, n_{500}, t_{500}), \\ (494, n_8, t_8), \ldots, (500, n_{14}, t_{14})$$

In this sequence, the triples are associated with the action entries. SESSION START entries are followed by a date entry. If 13 more action entries are recorded in the log, for a total of 533 entries (20 date entries and 513 action entries), the overflow will be detected by the AV-OS. Let 1 date entry and 20 action entries be overwritten. The printout of the log will show the following.

$$(21, n_{21}, t_{21}), (22, n_{22}, t_{22}), \ldots, (513, n_{513}, t_{513})$$

Effects: This defect can potentially limit the auditing capability to unambiguously determine the sequence of events that took place during the election and in fact allows a malicious manipulation of the log history. However, it should be clear that this weakness only manifests itself when a fairly large number of events are logged, a pattern which, in itself, can be an unusual state of affairs which can be easily detected. In any case, the process that led to the automation of the audit log was instrumental in uncovering a potentially serious defect in the logging module that could be maliciously exploited.

4.2 "Totals Report" Recording Deficiency

At the end of the election day the officials must close the elections and print the report with the results. The printing of this report is logged in the event log as the TOTALS REPORT action (Table 1). Prior election audits indicate that, sometimes, the event does not appear on the event log. Further investigation of the sequence of events that might follow the election closing reveals that several distinct sequences of interactions between the machine and its operator can lead to the same log trace. Specifically, we noticed that once we close the elections the machine proceeds to the printing of a copy of the totals report. Once the printout is complete the machine asks the operator whether s/he wishes another copy (as a yes/no question). If the operator replies negatively the machine proceeds and logs the action TOTALS REPORT and then prompts for a restart. If, however, the operator replies affirmatively (or does nothing at all) then the event is never recorded in the event log. Furthermore, while the terminal provides the capability of printing multiple copies of the election results, *at most one entry* for the printing action(s) is recorded in the event log.

Effects: No trace of printing the elections results may trigger a controversy, questioning the authenticity of the totals report, given than the event log does not confirm the printing. Moreover the inability to identify the number of reports printed may affect the auditing and electoral process. Note also that this behavior could be exploited to obtain a partial tally in the middle of the election by closing the elections, printing a totals report, never acquiescing to the subsequent yes/no question, proceeding to power-cycle the terminal and then resuming the election. Although the event log would contain a (benign) power-cycle sequence of events, there will be no record that a partial tally was produced.

More fundamentally, this constitutes a deficiency in the sense that there is no one-to-one mapping between the actual sequence of user actions and events in the log. Ideal logging should record every action and not limit itself to successfully completed actions or 'at-most-once' logging of repeated actions.

4.3 Date Recording Deficiency

The second deficiency directly relates to how date and time is recorded in the logged events. Recall that each action entry is a triple (s, n, t), where t is the time at which the named event n occurred. The time has the form `HH:MIN` where the hour field ranges from 0 to 23. As indicated before, the action entries INITIALIZE and SESSION START are always followed by a *date* entry. These are the only action entries that cause date entries to be recorded in the event log. The INITIALIZE action entry is logged only when the machine is programmed. The SESSION START action entry is logged whenever the voting terminal is power-cycled. To establish the time *and* date of an event one must obtain the date from the last preceding SESSION START event. Clearly, if the voting terminal is left turned on for more than 24 hours this method fails to yield the correct time stamp.

Effects: Under the current procedures established in Connecticut, this deficiency cannot be exploited for an attack. However, if the equipment is used with relaxed procedures where a district may upload their results to a central tabulator and avoid printing the actual election results, this may present a threat. Indeed, districts may keep their voting terminals on after the election, allow for more votes to be cast, then close the elections the next day at the expected time. In this case no audit procedure of the event log will be effective. On the other hand, printing the totals report enables the officials to check the close date on the printout.

5 Results of the Post-Election Analysis of Event Logs

Following the November 2008 election in Connecticut, we performed the analysis of the event logs collected from 279 AV-OS voting terminals used in the election. Additionally, 142 event logs were collected from the (de facto) back-up AV-OS voting terminals that were not used in the election. Thus in total 421 event logs were examined with our automated log analysis tool. The 279 collected event logs represent a random sample of over 30% of all precincts, thus the audit is broad enough to draw meaningful conclusions. (See our earlier work [2] on how the event logs are obtained from the voting terminals.) In this section we summarize our findings.

- The majority of event logs, 314 out of 421 appear to contain the expected sequences of events (see [1] for the procedures followed by poll workers in Connecticut).
- 15 event logs (3.6%) had more than 10 SESSION START events. This refers to voting terminal restarts. Some event logs had as many as four restarts in one minute.

  Among these event logs, 10 logs (2.4%) additionally had a COUNT RESTARTED event during the election day. This indicates that in some precincts there were difficulties with the terminals, perhaps ballot jams. It is suggested that records be made at the precincts in all cases where the terminals are restarted during an election to help diagnose any problems in the future.
- 41 event logs (9.7%) contained card duplication events with 5 logs indicating more than one duplication. Memory cards can be duplicated using a duplication procedure of the AV-OS terminal. This functionality is available in supervisor mode. There should be no reason to duplicate cards in Connecticut, and procedures clearly indicate so. One possibility is that duplication was performed when cards containing no valid data were discovered during testing. It is strongly recommended that duplication should not be performed at precincts and that all improperly programmed cards are reported to the Secretary of the State as soon as this is discovered.
- 29 event logs (6.9%) had a ZERO TOTALS REPORT printed before the date of the election. This can happen if one starts a voting terminal before the date of the election after setting it for elections.
- 24 event logs (5.7%) indicated that the corresponding memory cards were programmed from 10/27/2008 to 10/30/2008. While this is not necessarily problematic in the election, these cards were not subject to our pre-election audit that included only the cards programmed until 10/26/2008.
- 2 event logs had an additional ZERO TOTALS REPORT event during the election day. This happens if one restarts the terminal after finishing printing the zero totals report.
- 1 event log had ELECTION CLOSE event at 22:08. This stands out from the rest of the cards, however, it is not necessarily problematic as voters waiting in line are still allowed to vote after doors are closed.
- 6 event logs had the PREP FOR ELECTION event on the day of the election. Such event should have taken place some days before the election and not on the election day. This suggests that pre-election testing procedures were not followed.
- 4 event logs had a MEMORY CARD RESET event. Resetting the cards clears the counters after the voting terminal is placed in the election mode. All 4

event logs indicate that the reset was done before the election date, thus no actual votes were lost. Nevertheless, this indicates a deviation from standard procedures.

- 1 event log had an UPLOAD STARTED event. This indicates deviation from standard procedures (apparently the upload was attempted, yet there was no official receiving system). The corresponding card was not used in the election and otherwise the log appears normal.

- 2 event logs had test elections on 10/31/08 and 1 event log showed a test election on 11/03/08. We expected test elections to happen earlier.

- 1 event log has a test election on 11/26/08 and an election executed on 12/04/08. This suggests that the clock of the voting terminal was 1 month ahead.

None of the observations made in our analysis of the audit logs indicate a security problem or malicious intent. However, it appears that proper procedures are not followed uniformly. These results have been communicated to the Connecticut Secretary of the State, and as a result there will be amplification of election policies and procedures.

6 Conclusions

We developed an automated tool for analyzing event logs of the Premier AccuVote Optical Scan (AV-OS) voting terminal. We have used the tool to analyze the event logs from over 400 memory cards during the post-election audit of the November 2008 elections in Connecticut. This audit covered more than 30% of the precincts in the state.

The analysis of the event logs shows that there are some deviations from the prescribed and expected use of the optical scan terminals. In particular, we noted that 9.7% of the event logs contained memory card duplication events. The Connecticut Secretary of the State instructions to municipalities clearly do not allow for cards to be duplicated. Additionally, several event logs contained deviations from the expected relative timing of the events (although these deviations are apparently non-malicious). Other deviations included restart events (of benign nature) and events that indicated clearing of election counters (fortunately in all cases before the election).

The time-annotated rules developed for our event log analysis tool are specific to the AV-OS terminals and for the election procedures established in Connecticut. However, our approach is general and can be used to automate the analysis of logs produced by other voting machines under procedures specific to other states. This would require that (a) the rules are revised to reflect the events of another voting machine and the election procedures of another state, and that (b) logs are converted to an appropriate .xml format. Moreover, for the states that upload results for central tabulation, it should be possible to automate the extraction of the logs from the tabulation system and analyze the extracted logs.

Having performed and completed the post-election audit after the November 2008 Elections, we believe that voting terminals need to be periodically audited and that this is crucial in providing valuable information necessary to ensure the integrity of our electoral system. Our automated event log analysis did not reveal an immediate security concern or malicious intent. Nevertheless, our in-depth event log experimentation revealed a defect and two deficiencies of the AV-OS logging subsystem. Furthermore, the collected information from the actual election indicates that the proper procedures were not uniformly followed. We conclude that, despite minor deficiencies, event logs of voting terminals can be an invaluable auditing tool and we recommend that event log analysis be made a part of any post-election audit. We intend to continue our work on automated event log analysis and we plan to investigate the design of logging capabilities that would be rich enough to substantially increase the integrity of elections using them.

References

[1] A training guide for connecticut poll workers. Premier Election Solutions http://www.sots.ct.gov/sots/lib/sots/electionservices/training_info/ct_poll_worker_manual.pdf, 2008.

[2] DAVTYAN, S., KENTROS, S., KIAYIAS, A., MICHEL, L., NICOLAOU, N. C., RUSSELL, A., SEE, A., SHASHIDHAR, N., AND SHVARTSMAN, A. A. Pre-election testing and post-election audit of optical scan voting terminal memory cards. In *Proceedings of the 2008 USENIX/ACCURATE Electronic Voting Workshop (EVT 08), July 28-29, 2008, San Jose, CA, USA* (2008).

[3] DAVTYAN, S., KENTROS, S., KIAYIAS, A., MICHEL, L., NICOLAOU, N. C., RUSSELL, A., SEE, A., SHASHIDHAR, N., AND SHVARTSMAN, A. A. Taking total control of voting systems: Firmware manipulations on an optical scan voting terminal. In *Proceedings of the 24th Annual ACM Symposium on Applied Computing (SAC 09)* (2009).

[4] FELDMAN, A. J., HALDERMAN, J. A., AND FELTEN, E. W. Security analysis of the Diebold AccuVote-TS voting machine. http://www.usenix.org/event/evt07/tech/full_papers/feldman/feldman.pdf, 13 September 2006.

[5] Help America Vote Act. http://www.fec.gov/hava/law_ext.txt.

[6] HURSTI, H. Critical security issues with Diebold optical scan design. http://www.blackboxvoting.org/BBVreport.pdf, 4 July 2005.

[7] HURSTI, H. Diebold TSx evaluation. Black Box Voting Project, http://www.blackboxvoting.org/BBVtsxstudy.pdf, 11 May 2006.

[8] KIAYIAS, A., MICHEL, L., RUSSELL, A., SHASHIDAR, N., SEE, A., AND SHVARTSMAN, A. An authentication and ballot layout attack against an optical scan voting terminal. In *Proceedings of the USENIX/ACCURATE Electronic Voting Technology Workshop (EVT 07)* (August 2007).

[9] KIAYIAS, A., MICHEL, L., RUSSELL, A., SHASHIDHAR, N., SEE, A., SHVARTSMAN, A. A., AND DAVTYAN, S. Tampering with special purpose trusted computing devices: A case study in optical scan e-voting. In *Proceedings of the 23rd Annual Computer Security Applications Conference (ACSAC 2007), December 10-14, 2007, Miami Beach, Florida, USA* (2007), pp. 30–39.

[10] KOHNO, T., STUBBLEFIELD, A., RUBIN, A. D., AND WALLACH, D. S. Analysis of an electronic voting system. In *Proceedings of the IEEE Symposium on Security and Privacy* (2004), pp. 27–.

[11] NORDEN, L. The machinery of democracy: Protecting elections in an electronic world, 2005. Brennan Center Task Force on Voting System Security, http://www.brennancenter.org/page/-/d/download_file_36343.pdf.

[12] PAMELA SMITH, VERIFIED VOTING.ORG. Written testimony before the Committee on House Administration, Subcommittee on Elections U.S. House of Representatives. http://electionaudits.org/files/PamelaSmithTestimonyFinal_2007mar20.pdf, 20 March 2007.

[13] VORA, P. L., ADIDA, B., BUCHOLZ, R., CHAUM, D., DILL, D. L., JEFFERSON, D., JONES, D. W., LATTIN, W., RUBIN, A. D., SHAMOS, M. I., AND YUNG, M. Evaluation of voting systems. *Commun. ACM 47*, 11 (2004), 144.

[14] WAGNER, D., JEFFERSON, D., AND BISHOP, M. Security analysis of the Diebold AccuBasic interpreter. Voting Systems Technology Assessment Advisory Board, University of California, Berkeley, 14 February 2006.

[15] WERTHEIMER, M. A. Trusted agent report Diebold AccuVote-TS voting system. RABA Innovative Solution Cell, http://people.csail.mit.edu/rivest/voting/reports/2004-01-20%20RABA%20evaluation%20of%20Diebold%20AccuVote.pdf, January 2004.