Systemic Issues in the Hart InterCivic and Premier Voting Systems: Reflections on Project EVEREST*

Kevin Butler, William Enck, Harri Hursti†, Stephen McLaughlin, Patrick Traynor††, and Patrick McDaniel

{butler,enck,smclaugh,mcdaniel,traynor}@cse.psu.edu

†hursti@hursti.net ††traynor@gatech.edu

Abstract

The State of Ohio commissioned the EVEREST study in late summer of 2007. The study participants were charged with an analysis of the usability, stability, and security of all voting systems used in Ohio elections. This paper details the approach and results of the security analysis of the Premier and Hart systems within the EVEREST effort. As in previous studies, we found the election systems to be critically flawed in ways that are practically and easily exploitable. Such exploits could effect election results, prevent legitimate votes from being cast, or simply cast doubt on the legitimacy of the election itself. In this effort we identified new areas of concern including novel exploitable failures of software and election data integrity protection and the discovery of dangerous hidden software features. We begin by describing in depth our systematic methodology for identifying and validating vulnerabilities appropriate for the current complex political climate, and detail and illustrate broad classes of vulnerabilities uncovered using this approach. We conclude by considering the impact of this study both in terms of the tangible vulnerabilities discovered and as a model for performing future analyses.

1 Introduction

The Help America Vote Act (HAVA) of 2002 mandated the widespread use of electronic voting machines across the United States. As a result, the technology supporting elections nationwide changed nearly overnight. However, concerns about the security and integrity of elections conducted using available products arose nearly as quickly. As an increasing body of independent reports painted a bleak portrait of such systems, a number of state and federal officials began sanctioning official evaluations. Responding to declining public confidence in electronic voting machine technology the state of Ohio initiated an investigation of the risks associated with systems used across the state. Project EVEREST (Evaluation and Validation of Election Related Equipment, Standards and Testing) [1] brought together teams from academia and industry to develop a comprehensive understanding of the vulnerabilities and risks in the three systems used in Ohio: Premier Elections Solutions (formerly Diebold), Hart InterCivic, and Election Systems and Software (ES&S).

In this paper, we consider the results of the Hart and Premier analysis undertaken by the Pennsylvania State University academic team. Building upon past studies [2, 3, 4, 5, 6, 7, 8, 9, 10, 11], the team acquired voting systems used in Ohio and attempted to ascertain the presence and quality of the security provided. The team evaluated the degree to which the systems could provide for the integrity of elections. While some equipment had received prior analysis (e.g., the GEMS server), others did not (e.g., the Digital Guardian, the ExpressPoll electronic poll book, among others).

The broad results of the study were similar to those of the past–all studied election systems lacked the basic protections for guaranteeing election integrity. However, the results of the study differed to a large degree in their technical substance. In particular, there were several important failures detailed by this study that were not known prior to the release of this study. Several important discoveries include:

- *There is a veritable sea of previously undetected functionality in the Hart system. Note that we found what we believe is only a tiny fraction of the features enabled through undocumented software triggers, e.g., Windows registry entries.*
- *An attacker may subvert all back-end data protections in the Hart and Premier systems by exploiting*

*The comments made in this paper reflect the observations made during the course of our investigation and do not necessarily represent the opinions of the Secretary of State or the State of Ohio.

combinations of new and previously known vulnerabilities.

- *It possible to replace Hart system firmware in seconds with unfettered access to the equipment.*
- *The Premier EMP (memory card encoder) and VCE (voter token encoder) system components are vulnerable to a large number of previously unknown attacks that if exploited can undermine the entirety of the election process.*
- *The previously unevaluated ExpressPoll electronic poll book is trivially vulnerable to an attacker. The team was able to completely replace the software on the pollbook and compromise voter data within an hour of gaining access to the device.*
- *There exist critical failures in the previously unstudied Verdasys Digital Guardian security software. This software is used by the state of Ohio to defend the Premier GEMS server upon which the back-end election processes are based.*

The consequences of this study–as in others–are bleak. The flaws in the both the Hart InterCivic and Premier systems place the security of an election almost entirely on physical procedures. Our analysis suggests that when those practices are not uniformly followed, it will be practically impossible to detect attacks when they occur. Even in the unlikely event that an attack is identified, the resulting damage cannot often be contained and the public's belief in the integrity of the election restored.

The review team feels strongly that the continued issues of security and quality in both systems are the result of deep systemic flaws. Thus, we agree with previous analyses and observe that the safest avenue to trustworthy elections is to reengineer the Hart InterCivic and Premier systems to be secure by design.

The remainder of this paper discusses the execution and results of EVEREST study. We begin in the next section by describing how we systematically identified and confirmed each vulnerability. Thereafter, we demonstrate via example broad classes flaws and failures in the studied systems and discuss the potential impact of these vulnerabilities on voting systems and elections.

2 Methodology

Without an effective plan for evaluating these systems, conducting a truly comprehensive study is extremely difficult. Accordingly, while we believe the results of this study offer significant evidence of the systemic security problems with the Hart and Premier electronic voting systems, we expect future studies to follow. We therefore offer insight into our own methodology so that future researchers will be able to quickly and accurately evaluate such systems.

Key in evaluating any electronic voting solution is the understanding of that system's architecture. Simply understanding the components of a system is only the first step; more importantly, it is necessary to identify the relationships between components and understand both intended and unintended interactions. Previous evaluations of the same or similar equipment [2, 3, 4, 5, 6, 7, 8, 9, 10, 11] are an excellent means of bootstrapping this process. Our access to source code and equipment was severely time-limited, while contract negotiations and legal issues took up an enormous quantity of time at the beginning of the study period. Future studies will likely run into similar issues; thus we recommend that preparing as much as possible by thoroughly examining these previous studies. Because the particular configuration of an election system can vary between states, vendor-provided training is also recommended. We participated in day-long, high-level sessions on each system run by Hart and Premier. Combined, these approaches allowed us to understand how these systems are configured and operated in Ohio before examining a single line of source code or performing any red teaming.

When source code and equipment became available, they were voluminous in scope and quantity. We received dozens of pieces of equipment and substantial codebases–over 360,000 lines of code in the Hart system and over 330,000 lines of code in the Premier system. The sheer magnitude of the code, documentation, and number of components, coupled with the limited amount of time to perform the study, necessitated understanding system internals as quickly as possible. In the first phase of the study, we sought to independently confirm previous vulnerabilities for two reasons. First, locating such vulnerabilities allowed us to develop a more intimate knowledge of the system and understand critical interfaces and functions in the code. Understanding the kinds of problems known to be in these systems and the context in which they exist helped to point us to similar problems in previously unevaluated components. Secondly, by offering an independent evaluation of previous work, we provide more evidence to the public that such problems do in fact exist.

The EVEREST study was an opportunity in that the each team was given simultaneous access to source code and hardware. Accordingly, we were able to locate weaknesses in the software and demonstrate them on the machines themselves. Helpful to this process were tools such as a complimentary copy of Fortify SCA [12] and Doxygen [13], a freeware automated functional graph creator. Buffer overflows, authentication circumvention, and a wide variety of other attacks were then carried out against all of the evaluated systems. This procedure of

validation shows the ease with which many attacks can be executed, therefore we recommend future studies include similar validation in their evaluation to help combat "lack of real-world conditions" claims.

It is notable that while a tool such as a source code analyzer (e.g., Fortify) is very useful in limited circumstances, the vast number of errors that it reported across the entire codebase provided us with too much output to feasibly examine every condition. A further limitation of such a tool is that its close focus on individual routines, while useful for finding specific errors, is not initially conducive to understanding broad designs of the system architecture. Understanding these details required a more comprehensive and analytic approach. To this end, we focused intently on inputs to components such as the user interface and closely examined cryptographic APIs and structures to understand how secure information was handled. This was critical for the second phase of our study: determining new vulnerabilities within each system and examining the new equipment supplied by Premier.

Finally, we recommend that future evaluators replicate our procedure of creating a detailed unredacted description for *every* vulnerability, independently confirmed by another member of the team. More precisely, our issue discovery and confirmation process proceeded as follows.

1. Identify a potential vulnerability or area of concern.
2. Perform a detailed source code analysis and/or exploit the vulnerability.
3. Write a detailed description of the vulnerability including enough information to replicate the experiment.
4. Acquire independent confirmation from a team member not involved in the discovery of the vulnerability.

Only after independent confirmation could a vulnerability be included in the report. The documentation played a key role in the process, because it allowed us to convince the rest of the team that a vulnerability exists. Possibly more important, it allows future analyses and third parties to recreate our work. As such, the descriptions should contain line numbers, code samples, and file and function names. While the previous studies contained private portions for some vulnerabilities, lack of access to such information for all vulnerabilities required our team to spend significant time in the confirmation phase. Note that we were given limited access to the private appendices for some of the previous Premier studies; however, it occurred during the closing days of the study and therefore provided minimal value. No such information was provided for the Hart Systems. As a result, we spent many hours in some cases understanding esoteric and obscure code structures, often scattered amongst dozens of files, to track down what often turned out to be minor pieces of information that were nonetheless essential for confirming a vulnerability. Having access to the reports detailing how to find some of these vulnerabilities would have led to faster confirmations and led to faster understanding of the system, saving a large amount of time. To avoid these problems, future similarly sanctioned studies must be given unregulated access (under appropriate nondisclosure agreements) to private reports containing specifics for each vulnerability at the *beginning* of their evaluation to ensure that the majority of the study can be spent on previously unevaluated components.

Our assessment methodology was particularly effective - in nine weeks, this study doubled the number of publicly known vulnerabilities in Premier systems and found over 25 new vulnerabilities in the Hart system. In fact, as the evaluation approached its end, the rate of vulnerability discovery continued to *increase*. Given more time, it is our firm belief that additional significant vulnerabilities would continue to be found. By structuring future studies in a similar manner, we believe that even more comprehensive evaluations can be carried out successfully.

3 Evaluation Results

Detailed at length in the followings sections, the vulnerabilities discovered as part of this study are representative of universal classes of flaws shared by the studied systems. The flaws include:

Failure to effectively protect election data integrity: Virtually every ballot, vote, election result, and audit log is forgeable or otherwise manipulatable by an attacker with access to the voting systems. Further issues expose voter choices and can lead to voter coercion and vote selling. These vulnerabilities place enormous burdens on the physical procedures of an election. Examples of these vulnerabilities are given in Section 3.1.

Failure to protect an election from malicious insiders: Neither system provides adequate protections to ensure election officials, poll workers, or vendor representatives do not manipulate the system or its data. These attacks are often invisible after the fact, and therefore misuse is difficult or impossible to uncover later. Examples of these vulnerabilities are given in Section 3.2.

Failure to provide trustworthy auditing: The auditing capabilities of the Premier and Hart systems are limited. Those features that are provided are subject to a broad range of attacks that can corrupt or erase logs of election activities. This severely limits the ability of election officials to detect and diagnose attacks. Moreover, because

the auditing features are generally unreliable, recovery from an attack may in practice be enormously difficult or impossible. Examples of these vulnerabilities are given in Section 3.3.

Unsafe features and practices: The studied systems embrace dangerous designs and practices. Each system possesses undocumented features that are highly dangerous. Further, the visible lack of sound engineering practices leads to widespread security and reliability failures. Examples of poor or unsafe coding practices, unclear or undefined security goals, technology misuse, and poor maintenance are pervasive. This general lack of quality leads to buggy, unstable, and exploitable systems. Examples of these vulnerabilities are given in Section 3.4.

The remainder of this section revisits each of the failures and highlights different examples of them in the studied systems. Readers not familiar with the terminology and operation of the Hart and Premier products are directed to Appendices A (Hart) and B (Premier) for brief primers on these systems.

3.1 Election Data and System Integrity

Attack Class 1: MBB Image Manipulation – Hart systems use a mobile ballot box, or MBB, as the main conduit for recording and tabulating votes. The data on an MBB, also known as its "image", is stored on a PCMCIA card. At a low level, it is a persistent storage device that can have data removed from it by simple copying (e.g., using the UNIX `cpio` utility). An attacker can use this feature to manipulate an election. If the attacker can access the MBB and copy data from it, then come back later and write the copied data back to the MBB, all of the votes cast in the period after the MBB was initially replaced will be effectively erased (EVEREST, 20.1.1). From the MBB's standpoint, there is no indication that any votes occurred during this time period, with only the internal logs of the precinct equipment (the eScan or the JBC and eSlate, depending on whether optical-scan or DRE voting is used) maintaining this record. Because the MBBs are used for the final vote tally, unless a recount is performed, the chances of the missing votes being caught are small. This attack was briefly alluded to in the California red team report [9] but was non-specific in terms of the threat or the attack.

Attack Class 2: Casting an Unlimited Number of Ballots – Premier Election Systems use smart cards to ensure that each voter is only able to cast a single ballot per election. After casting their ballot on an AV-TSX, the card reader marks the card as "Cast". If this card is reinserted into an AV-TSX before it is re-enabled by a poll worker (using either the ExpressPoll or Voter Card Encoder), the voting machine ejects the card and alerts the user that it has already been used. Implemented correctly, this mechanism should prevent a single user from casting more than their allotted single ballot.

Using multiple vulnerabilities discovered during the EVEREST evaluation, it is possible to enable a voter to bypass these mechanisms and cast an unlimited number of votes. Moreover, the evidence of such an attack can be erased. Worse still, this attack requires no special tools or private knowledge of the system.

We assume that our attacker approaches the voting booth during an election under normal circumstances. The attacker brings with them a stack of smart cards containing the default Smart Card Key (published on the Internet). After approaching the AV-TSX, the attacker begins by covering his/her tracks. Because the AV-TSX notes in its audit logs when cards have been encoded, the attacker accesses the Central Administrator mode by exploiting EVEREST Issue 14.8.7. Here, the attacker can delete the contents of both the memory card and the AV-TSX, thereby erasing most evidence of the attack. To hide the card creation operations, the attacker then simply changes the time and date of the AV-TSX to a period before the election. This portion of the attack can be accomplished in just over one minute. Moreover, deleting the contents of the memory card and changing the time/-date are not logged. Should the attacker also worry about the log information encoded on the VVPAT, weaknesses in the enclosure allow the paper record to be rendered unreadable (EVEREST, Issue 14.8.3).

To encode voter cards, the attacker gains access to the Supervisor Menu by exploiting the vulnerability described in EVEREST Issue 14.8.8. Access to this menu can be achieved consistently in less than one minute. The attacker then encodes the stack of smart cards smuggled into the voting precinct as valid Voter Cards, each of which takes a few seconds. There is no limit to the number of cards that can be programmed.

The attacker can then walk away from the machine and give the cards to colluding adversaries in the parking lot. These adversaries can use the cards to cast extra votes.

Attack Class 3: Exposing Voter Choices – Premier Election Systems recently introduced the ExpressPoll to replace traditional paper voter log books at the polling place. While the ExpressPoll contains various vulnerabilities allowing files and software to be manipulated (see Section 14.6 of the EVEREST report), a nontrivial privacy concern results from the technique used to audit the device. Note that the ExpressPoll does not directly participate in vote tallying; however, it encodes Voter Cards, and therefore the audit log should indicate if a voter's status was reset to allow multiple votes.

When voters enter the poll place, they are authenticated via information present in the ExpressPoll. Once authenticated, the voter is given a Voter Card, and the

voter information database is updated to indicate the voter has already entered the polling place. Along with this update, the ExpressPoll appends the activity audit log indicating the `voterId`. The `voterId` field recorded in the audit log matches a similar field in the voter information database. As the audit log is appended, the order voters enter the polling place is captured with a sequence number, and while a timestamp is not recorded for these entries, other entries, e.g., power-on, include a timestamp (EVEREST, Issue 14.6.7). Hence, an attacker can derive approximate times for voter entries. The voter order can also be correlated with VVPAT records for the AV-TSX to determine with some probability each voter's choice. Such information enables vote coercion and places significant tension on the efficacy of the election process.

Attack Class 4: ExpressPoll Software Integrity Failures – In order to allow updates to the bootloader and operating system, the ExpressPoll scans all inserted memory cards (both PCMCIA and CF) on boot. If the bootloader finds a file purporting to be a new bootloader (EBOOT.BIN) or Windows CE (NK.BIN), it erases the previous version of the software and loads the new version from the above file(s). Like the vulnerabilities previously discovered by Hursti, at no time is the source of these files authenticated; rather, a file on the memory card with either of these names will automatically be loaded and executed. Accordingly, anyone that can power cycle an ExpressPoll and insert a new memory card (i.e., any poll worker) can exploit this vulnerability. This vulnerability exactly mirrors Hursti's report on the AV-TSX, except that it has been re-implemented in a new system.

We note that there is a chance that placing the Windows CE file (NK.BIN) will not replace the current operating system, but rather only boot from it. Due to the potentially destructive nature of the testing and the fact that we were not given builds or most of the source code for the ExpressPoll, we verified that the files are accepted, but did not allow the process to be completed. Regardless, either set of functionality, booting as a runtime image or direct flashing, offer the same potential. Once booted, the runtime image can flash itself to permanent memory.

These vulnerabilities are one of a number of ways by which an adversary can gain access to the database of eligible voters. Accordingly, voters could be arbitrarily added or removed from such a list, thereby potentially compromising the integrity of the election and or disenfranchising voters.

Attack Class 5: VCE Software Integrity Failures – In order to allow for software updates, the VCE can be reprogrammed using a 9-pin serial cable attached to a PC. To load new software onto the VCE, a user simply turns the device off. When the user presses the off button again, the Voter Card Encoder prompts the user to press the "Yes" button if they would like new software to be loaded.

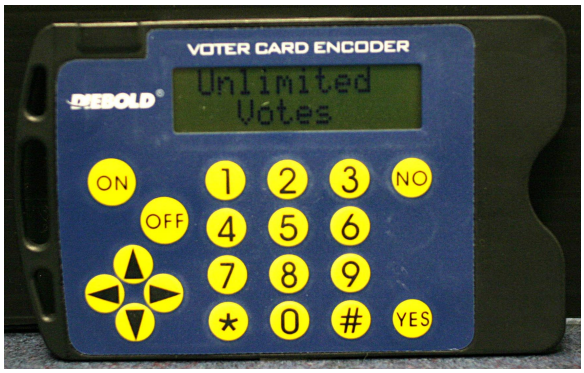

Figure 1: A VCE running arbitrary software.

The problem with this update mechanism is that it lacks any authentication of the new software loaded onto the VCE. As a demonstration of this issue, we created and loaded new software. Where the software provided by the manufacturer requires a user to activate the VCE with a Supervisor Card, we allowed any card (even unrecognizable formats) to enable the device. An adversary could therefore steal a VCE, load their own software and then create valid Voter Cards. Figure 1 shows an example of our modified software created in Issue 14.5.3 of the EVEREST report. Alternatively, we could have used this vulnerability to encode any type of card we wished. For instance, an attacker could easily create Central Administrator, Security or Supervisor Cards by further modifying the software running on the VCE.

3.2 Insider Defenses

Attack Class 6: eScan Manipulation – We were able to exploit a number of vulnerabilities in the eScan that could give election insiders the ability to compromise election results and voter privacy. Some of these were a result of a lack of physical security. We were able to replace the eScan's internal flash memory card containing the eScan executable and configuration file with only a screwdriver in about 2 minutes. After replacing the card, we were able to boot the eScan into the Linux operating system. This simple attack gives a single poll worker with a few minutes of unobserved access to the eScan the ability to undermine all votes cast at a precinct (EVEREST 20.3.1).

While opening the eScan to replace the memory card, we broke three tamper evident seals. While such seals may prove that a machine was opened, a preventative

measure is preferable. A poll worker may intentionally break these seals in order to cast doubt on election results. It has also been shown that tamper evident seals do not always correctly show that tampering occurred [14].

Insiders may also wish to use their access to ballots to determine voter choice. This can be done with the eScan due to the design of its ballot box (EVEREST 20.3.4). The eScan's scanner sits on top of its ballot box, which is essentially a plastic tub. When a ballot is scanned, it is subsequently dropped into the box. No measures are taken to disturb the order in which ballots fall, allowing a malicious poll worker to note the position in which certain votes are cast and then relay these positions to an election official with access to the ballots. We observed ten numbered ballots as they were cast with the eScan, and verified that the vote order was preserved.

Attack Class 7: JBC Manipulation – Normally, the voter access codes needed to vote using an eSlate are generated by the JBC and printed. It was previously shown that these voter codes could be rapidly generated from the JBC's serial port during the early voting phase of an election (Issue 4, CA TTBR). This was accomplished by disabling the JBC's printer through the menus. Rapid generation of voter codes allows a poll worker to collude with voters to vote multiple times. In our investigation of this vulnerability, we found that contrary to initial findings indicating that the maximum number of outstanding access codes was 150, we were able to generate over 10,000 access codes within an expiration period (set to 30 minutes by default, but configurable to as high as 16 hours) [15] This ballot stuffing attack is limited however, in that a large number of votes during early voting would likely be conspicuous and easily identified as fraudulent. For this reason, we investigated ways to rapidly generate voter codes during the normal voting period.

It is not possible during the regular voting period to disable the JBC's printer through its menus. We discovered that requesting an "Access Code Report," over the serial interface while there was no paper in the printer re-enabled the menu option to disable the printer. Once this option is available, the printer can be disabled and voter codes can once again be generated rapidly (EVEREST 20.4.1). This is an example of bad exception handling, which is seen elsewhere in the Hart system, such as in the case when a user database is empty allowing the creation of administrator accounts.

Attack Class 8: EMS Manipulation – The Hart system places nearly complete trust in the physical security and the procedures at election headquarters. The lack of security in the Hart components located at election headquarters is in direct conflict with the total power that election officials have. One of the most crucial components of the back end system is Tally, the vote tallying software. Improper use of Tally can lead to partial or total corruption or loss of election results.

Tally maintains a database containing the state of all MBBs used in an election. If an MBB is marked as tallied in this database, Tally will refuse to count the results on that MBB. Thus deliberate or accidental tallying of an MBB by a poll worker can lead to the results on the MBB not being counted. Note that because the state of the MBB is stored in the database and not the MBB itself, a malicious election official could mark MBBs as tallied by manipulating the database (EVEREST 20.6.1).

A unique feature of Tally among the EMS components is that its user interface is completely configurable through the Windows registry. Each registry entry specifies the DLL used to implement the behavior of a certain UI component. Modifying these behaviors in the registry can lead to subtle errors that are hard to detect (EVEREST 20.6.2). For example the import MBB and export MBB dialog boxes are exactly the same with the exception of one word. Unless the EMS systems are reinstalled and reconfigured between elections, which is highly unlikely, an election official could introduce such errors to Tally that would affect future elections. Such actions are nearly impossible to trace.

Attack Class 9: Circumventing Digital Guardian – Digital Guardian was designed to protect a system running Windows 2000 or XP. It allows an administrator external from the local system to specify policies that control how all local users are allowed to execute programs and access files. In Ohio's setup, a state employee possesses a special laptop called the Digital Guardian console. Each GEMS server contains the Digital Guardian Agent that enforces the policy specified by the console. The only way the Digital Guardian Agent can be disabled is if a state employee directly connects the Digital Guardian console to the GEMS server and specifies that the agent should be disabled. Our analysis of the Digital guardian yielded three categories of vulnerabilities: configuration flaws, means of disabling Digital Guardian, and flaws in the Digital Guardian software itself.

The Digital Guardian configuration contains a number of addressable flaws. One of the more significant enablers for circumventing Digital Guardian is the configuration of Microsoft Windows. Specifically, the *GEMSUser* user account is in the Windows *Administrators* group (EVEREST, Issue 14.7.2). Many of the deeper vulnerabilities we discovered rely on administrative access, which can be assumed given this configuration. The Digital Guardian policy itself also contained simple misconfigurations. For example, the Nero CD burning application can rename GEMS database files (EVEREST, Issue 14.7.8) thereby allowing an attacker to modify its contents before replacing the original. In both cases, configuration fixes could mitigate the vulnerabilities.

Deeper configuration errors stemmed from limitations of the general approach for policy specification. The policy "blacklists" specific potentially dangerous applications (EVEREST, Issue 14.7.6), e.g., the registry editor. Blacklisting has a fundamental limitation: it cannot practically identify all current and future applications. For example, we used a command line task scheduler to launch a shell as the "SYSTEM" user, which bypasses all Digital Guardian protections (EVEREST, Issue 14.7.5). Furthermore, one blacklist identification technique relies upon the cryptographic hash (MD5) of the application, thereby allowing an attacker to circumvent protections by simply modifying one bit in the binary.

The limitations of the blacklist policy more fully manifest in techniques to disable Digital Guardian all together. While BIOS passwords help prevent an attacker from booting from external media to disable Digital Guardian (EVEREST, Issue 14.7.3), the policy failed to blacklist access to `C:\ntldr`, which defines the location of the boot loader configuration (`C:\boot.ini`), a file specifically blacklisted by the policy. Hence, an attacker can modify `C:\ntldr` to use a different file, e.g., `C:\b00t.ini`, that is under the attacker's control (EVEREST, Issue 14.7.1). By modifying the boot loader configuration, Grub4DOS can be used to boot from a CD-ROM and disable Digital Guardian. Additionally, the policy did not blacklist "Device Manager," which we found can be used (only once) to disable the device drivers implementing the Digital Guardian enforcement mechanism (EVEREST, Issue 14.7.4).

A final category of discovered vulnerabilities were unrelated to the configuration. Rather, we believe them to be flaws in the Digital Guardian implementation. While we were not provided technical documentation from Verdasys, experience with similar tools brought us to the conclusion that when the policy identifies an application by a cryptographic hash (e.g., for blacklisting to deny execution) the enforcement mechanism should calculate the application's hash on demand (e.g., if an application is blacklisted from executing, every time an application executes, the hash should be calculated and compared against those in the blacklist). Contrary to this expectation, we successfully executed blacklisted applications by copying them to a new location (EVEREST, Issue 14.7.7). While we were unable verify the exact enforcement technique, we speculate that Digital Guardian caches a table mapping file paths to hash values, and the file path identifies applications, leaving the system susceptible to a TOCTTOU attack.

3.3 Auditing

Attack Class 10: EMS Audit Log Manipulation – Many Hart EMS applications (BOSS, Ballot Now, SERVO and Tally) maintain audit logs of the functions they have performed. These logs are stored in databases, with every entry including a date and time when an action was performed, the name of the user performing the logged action, a numeric identifier for the action (the pairing of this identifier and its verbal description are located in another database table), and data pertaining to the log entry (e.g., an adjusted vote total).

The database storing the audit log may be accessed by an unprivileged attacker and the logs modified such that any evidence of tampering in the voting system is covered (EVEREST 20.1.4). This can be done by first extracting database passwords from application configuration files, as detailed in Issue 15 of the CA TTBR. We used a freeware software utility that allowed us to communicate to the database through an ODBC interface and issue SQL commands directly. We were able to perform arbitrary operations on the databases in this manner. For example, an operation in Tally allows for the manual changing of vote totals; we were able to remove the audit log entry for this operation, or modify it to reflect an innocuous operation instead by changing the numeric identifier for the action.

Attack Class 11: EMP Log Manipulation – The Premier EMP server, which is responsible for the parallel reading and writing of memory cards used in the AVTSX, keeps logs of many of the operations it performs. For instance, when a blank memory card is inserted and a new ballot definition downloaded, the EMP server creates a log entry. Logging also occurs when cast ballots are uploaded to GEMS or when an error (e.g., connection timeout) occurs. These logs provide evidence with which an auditor can reconstruct the events on an election.

Like the ExpressPoll, the integrity of the EMP logs is not protected. During the course of our investigation, we were also able to alter entries from outside of the application, and then properly view them in the EMP's log screen. By escalating our privileges in the operating system, we were able to simply erase such files without raising any alarms. Instead, like the ExpressPoll, the EMP simply created new log files when old ones were deleted. This vulnerability was reported in Issue 14.1.5 of the EVEREST report.

The EMP is, in most configurations, the gateway between back-end processing and all touchscreen voting machines throughout the county. Accordingly, events taking place on this platform are extremely valuable to recreating an election. For instance, if a virus were to spread from a precinct to the central headquarters, as was suggested in a number of previous works [6, 16], logs at the EMP would be a valuable tool in identifying the source of such an attack. However, such mechanisms are of limited value to any post-election audit as their integrity simply cannot be trusted.

Attack Class 12: eSlate VVPAT Manipulation – In Ohio, the Hart eSlate DRE machines are used in conjunction with VBO printers that produce a verified-voter paper audit trail (VVPAT), with the resulting generated paper record acting as the legal ballot. The eSlate controls the VBO through a 1/8-inch port that is accessible by removing the VBO from its housing. The eSlate housing has a large black release button above the VBO, allowing it to be removed. The accessible port is the interface through which a variety of operations to the VBO are performed, including sending messages to be printed, checking whether the printer is low on paper, setting the VBO's serial number, printing debug information, and checking for general printer error conditions. There is no authentication of commands that arrive over this interface. As a result, an adversary who can control the interface to the printer can print arbitrary data to it, as described in Issue 34 of the CA TTBR. Notably, other interfaces may lead to the sending of privileged commands to the VBO. In particular, the serial number may be changed through the parallel port of the JBC and the eSlate's serial port in addition to using the 1/8" VBO port; we successfully changed the VBO's serial number using the JBC's parallel port by writing a short C program on a laptop and attaching it to the JBC. A modified serial number could call into validity the votes recorded to the VVPAT (EVEREST 20.5.5).

The VBO printer is easy to disable. The VBO connects into a power cable and a data cable. If either of these is severed, particularly if it is done skillfully, then the connected eSlate will show a communication error that is hard to diagnose. Since the VBO is not field-serviceable, a new one would need to be brought in and determining the core problem may be difficult. The eSlate can hence be knocked out of service for a significant amount of time, perhaps the duration of the election, potentially causing voter disenfranchisement. The eSlate takes approximately 15 seconds to report an alarm to the JBC, leaving ample time for an attacker to leave the polling place before malfeasance is suspected (EVEREST 20.5.2).

The VBO may potentially be handled by the voter, as a large black button on the eSlate's housing allows the unit to be removed, though it is not meant to be handled in a polling place. The back of the VBO has a pair of screws that may be turned by hand to access the interior of the unit. The paper may then be removed from the spools and either replaced or the reattached after removing the portion of the roll on the take-up spool (EVEREST 20.5.4). We found it was possible to perform this in as little as one minute, with the movements obscured by the privacy shield attached to the eSlate housing. However, the JBC's LED for the eSlate may flash when the data cable is detached from the VBO, although it is possible with care to perform the operation without causing the JBC to flash.

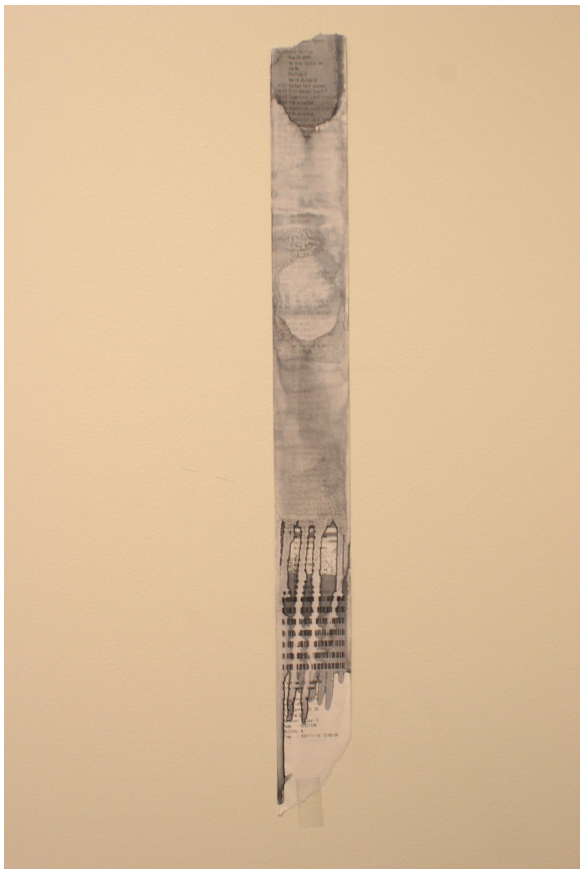

Figure 2: A printed system log destroyed by injecting a household chemical into the AV-TSX. No tamper-evident seals were broken or disturbed in the attack.

Even if the VBO is not itself compromised, there is little assurance that the generated VVPAT is trustworthy. When the VBO prints the accepted vote, a two-dimensional barcode is printed in the standard PDF-417 format, making it easy to generate. The rest of the ballot is generated in plain text, as alluded to in the CA TTBR. Nowhere is any authenticating information (such as an HMAC) embedded into the barcode or printed anywhere else on the ballot. As long as an adversary knows the serial number of the VBO, an entire roll can be forged and either replaced in the VBO (an operation that can take about a minute in a precinct) or when the tape from the VBO is removed (EVEREST 20.5.5). It is not clear whether the bar code is used to tabulate results from the paper roll or whether it is examined at all.

Attack Class 13: AV-TSX VVPAT Manipulation – The paper audit trail generated by the AV-TSX machines operated in Ohio is cited by many as a failsafe means of recording a voter's intent. Before a ballot is cast, each

voter is afforded the opportunity to evaluate a printout of their selections and, should the electronic count be disputed, election administrators can rely on these receipts as the official legal record of the ballot. Unfortunately, the VVPAT system used by these machines is poorly constructed and subject to a number of attacks that negate their perceived value.

Chief among these problems is the construction of the VVPAT system itself. Protected by a thin and flexible plastic enclosure, the physical security of the printer is a significant risk in the Premier system. For instance, as discussed in Issue 14.8.1 of the EVEREST report, the wires connecting the printer to the AV-TSX can easily be exposed by pushing the edge of the plastic covering. An adversary wishing to disable the printer on an AV-TSX could simply cut these wires without breaking any tamper-evident seals on the device. Alternatively, the plastic housing itself can simply be removed from the AV-TSX with minimal physical effort. Issue 14.8.2 of the EVEREST report notes that this enclosure is attached to the AV-TSX by a 1/8 inch plastic latch. By applying the appropriate pressure, an adversary can gain access to all previously cast votes without raising significant attention. Should the results of an election be disputed, the absence of a paper trail from these attacks would prevent a complete recount from occurring.

An attack less likely to catch the attention of poll workers until the close of an election is possible because of the inadequate sealing of the printer enclosure. As discussed in Issue 14.8.3 of the EVEREST report, an attacker can exploit this weakness by using a syringe to inject a common household substance known to degrade/destroy information written to thermal printer paper. Such a compound could be inserted in multiple ways such that all previous paper ballots stored in a machine would become unreadable. Alternatively, all of the unused paper in the machine could be attacked, preventing all future votes from being tallied. An example of the first attack is shown in Figure 2. Note that the results of the audit log are unreadable. A similar vulnerability was discussed in the California Red Team report (Issue 4.f) [8]; however, the details of this vulnerability were not listed in the public report.

A number of factors of the AV-TSX VVPAT system combine to make such attack possible. The use of an inexpensive and pliable plastic enables the first two attacks. Thermal printers, of which the use for creating long-lived records is recommended against due to fading problems, enable the latter. Because of these weaknesses in implementation, the VVPAT results generated by an AV-TSX cannot be relied upon as the only auditing mechanism for Premier systems. Unlike more traditional systems in which ballots are kept in a central, guarded ballot box, VVPATs simply do not provide the same protection of a voter's intent.

Attack Class 14: Open Audit Interfaces – Both the Hart JBC and eScan have open interfaces that allow for the erasure of votes and audit log records. As detailed in Issue 3 of the CA TTBR, the eScan is managed through an accessible Ethernet port that listens for connections on TCP port 4600. This port is normally used for sending and receiving commands from SERVO, such as file transmission and reading images of the eScan's memory. No cryptographic tokens are required for these operations to occur.

We discovered that with a handheld device such as a Palm computer, an attacker with an Ethernet cable can mimic the actions of SERVO to the eScan during a live election, and cause the vote records and audit logs to be erased from both the eScan's internal memory and the MBB inserted into it (EVEREST 20.3.7). Any voting that had occurred on the eScan to that point would be erased, necessitating a manual recount.

The JBC is similarly vulnerable to attack (EVEREST 20.4.2). SERVO connects to the JBC over a parallel port interface. If a Palm handheld with a parallel port interface is connected to the JBC, it may be used to clear the vote records and audit logs from the JBC's internal memory and the MBB attached to it. Since the JBC controls the eSlates as well, it is also possible to clear their vote records and audit logs from the JBC's parallel interface. We wrote a program that allowed us to reset the JBC and eSlate from a laptop and found that all evidence of voting on that machine had been cleared.

3.4 Unsafe Features and Practices

Attack Class 15: Password and Key Misuse – The Hart EMS applications BOSS, Ballot Now, SERVO, and Tally, require a username and password to log in. These credentials are stored in a security database associated with each application. We were able to connect to the database though an attack described in Issue 15 of the CA TTBR, where the database passwords are kept in configuration files that are easily read. At this point, we can delete the usernames found in the database. Once this has been done, the applications may be opened and a new administrator account created. The application can then be logged into with administrative access (EVEREST 20.1.3).

With supervisor access to these applications, it is possible to modify the processes of ballot definition and creation, tallying of votes, and maintenance of the voting equipment. Ballots may be arbitrarily printed in Ballot Now, and the audit logs for voting equipment may be cleared.

The Ballot Now application contains an additional

password-based vulnerability. Ballot Now connects to a back-end Sybase database, which runs a stored procedure when a user logs in, taking a hash of the username and password as input to be validated. By replacing the stored procedure definition, found in the security database, with a single line of code, we were able to allow any user to log into Ballot Now with any username and password, or with none at all (EVEREST 20.7.1).

Key management problems are well known in Premier systems. As was demonstrated in the CA TTBR (Issue 5.2.5), keys are insufficiently protected in units such as the AV-TSX. However, our analysis of the EMP uncovered additional problems in the key management of such systems. As conjectured in the CA TTBR and confirmed in Issue 14.1.2 of the EVEREST report, the Data Key used to protect the results of an election is the same in every machine in a county. Key management in the EMP is substantially more dangerous. As discussed in Issue 14.1.7, the System Key used to encrypt the Data Key is derived from the system's serial number. Like the System Key in the AV-TSX, the System Key for the EMP server is created in a predictable manner. The machine's serial number is fed as input to the MD5 hash algorithm, the deterministic result of which becomes the System Key. On each AV-TSX, this serial number (and therefore the resulting System Key) is unique. However, the serial number used is a fixed value on all machines: 0. Accordingly, every EMP server created uses the same System Key. Such key management strategies fail to provide containment against compromise and therefore allow a successful attack on a single machine to potentially compromise elections on a large scale.

Attack Class 16: eScan Functionality – One example of unsafe functionality being seamlessly added to a necessary interface is the eScan's configuration file. This file can be retrieved and uploaded via the eScan's Ethernet port, as described in Issue 3 of the CA TTBR. The protocol used to communicate over this port is simple and has no facilities for authentication between the eScan and any host to which it is connected. The configuration file is obtained by issuing a single numerical command to the eScan, and uploaded by issuing a similar command and sending the file. We wrote programs to do both using standard sockets APIs.

The default configuration file contains an option to "allow duplicate ballots", which is commented out. We uncommented this option and uploaded the file. We then carried out an election using photocopies of a single filled in paper ballot. With the option enabled, the ballots were accepted by the scanner and the vote totals stored to the MBB (EVEREST 20.3.6). These votes were counted and reported on the eScan's paper printout and were tallied by Tally. Note that without enabling the duplicate ballots option, any copy of a paper ballot is rejected by the scanner after the first instance is scanned. Along with the photocopied ballots, we were also able to attach a piece of tape to a single ballot and retrieve it from the eScan after scanning, allowing us to vote multiple times with a single ballot, albeit in a more conspicuous manner than with photocopied ballots (EVEREST 20.3.9).

It is still possible however, to detect that multiple duplicate ballots have been scanned. The eScan's audit log contains the serial number of every ballot scanned, allowing a vigilant auditor to uncover the duplicate ballots. This could be avoided with the assistance of a malicious poll worker erasing the eScan's audit logs at the polling place. Even if the audit logs are deleted, the duplicate ballots can be discovered by examining the bar codes on each paper ballot in the ballot box. This too is undetectable if the above approach of retrieving a scanned ballot is used.

We also discovered an undocumented telnet server running on the eScan (EVEREST 20.3.2). The server is the Microsoft Windows CE Telnet service. Most likely, the server started by default, suggesting a lack of proper configuration of the underlying OS. While we were not able to login to the telnet server, vulnerabilities have been discovered in other Microsoft telnet servers [17, 18], indicating that it may be possible to gain control of the eScan by exploiting the server. While disabling the server may easily mitigate this issue, the extent of the misconfiguration of the OS underlying the eScan software remains unknown.

Attack Class 17: JBC and eSlate Functionality – The eSlate and JBC also have a significant amount of unsafe and undocumented features integrated into their standard functionality. The most outstanding of these is the ability of the JBC to receive and issue "soft" button presses (EVEREST 20.4.3).

These are button presses not created by the actual buttons on the JBC or eSlate, but encoded in a communication protocol. The JBC receives these soft button presses via its parallel port and can forward them an attached eSlate via its serial port. Upon receiving a soft button press, the JBC will decide whether to process it or relay it to an attached eSlate.

When a device receives a soft button press, it first makes a call to the underlying OS to insert the button press as a regular keyboard interrupt. The OS then delivers the keycode to the application for processing. This method of delivery makes it impossible for the keyboard input processing components of the JBC and eSlate to determine whether a button press is from the keyboard or an external device.

Using the soft button press functionality, we carried out a "Ghost Voting" attack on the JBC and eSlate. This attack allowed us to connect a laptop to the JBC's parallel port and automatically vote for selected candidates

an arbitrary number of times. The laptop was running a program we wrote that works as follows:

1. Obtain a voter code from the JBC's parallel port.
2. Enter the voter code into the JBC by sending soft wheel turns over the serial cable connecting the JBC to the eSlate.
3. Send the appropriate soft button presses and wheel turns to the eSlate to vote for the desired candidates.
4. Complete voting and approve the VVPAT
5. Repeat

This program contained approximately 200 lines of new code, and required slightly over two hours to complete. With it, we were able to enter a registration code, vote and approve the VVPAT once every 20-30 seconds. Note that no authentication was required to send the soft button presses. Each vote was recorded on the eSlate's VVPAT, the JBC's unofficial printout and the cast vote records stored on the JBC's MBB. These vote records were tallied by Tally and there was no evidence in the audit logs suggesting that malicious behavior had occurred. Along with the soft button presses, step 1 of our program also relied on the ability to generate voter codes via the JBC's parallel port.

Attack Class 18: EMS Key Misuse – Another example of undocumented and unsafe functionality is the ability of the Hart Election Management System (EMS) applications (BOSS, Ballot Now, Tally, SERVO and eCM Manager) to silently write all or part of the eCM key to a debug file in cleartext (EVEREST 20.2.1). By silently, we mean without any notification through the user interface that the key will be stored.

This functionality is not a part of the EMS applications proper, but of the Spyrus library they use to read and write the eCM tokens, which are Spyrus Rosetta USB tokens. When any EMS application reads the key from the token, the Spyrus library checks a specific entry in the Windows registry for a path to a debug file. If this entry is found, 16 out of 40 bytes the key are saved to the debug file in plaintext. When the eCM manager writes the key to the token, the Library writes the entire 40-byte plaintext key to the debug file. An attacker with very brief access to an EMS system could enable the Spyrus registry entry and later check the contents of the debug file to obtain the county wide key.

Attack Class 19: Autovoting – A final example of unsafe features intentionally added to the Hart systems is the Ballot Now's "Autovote" feature (EVEREST 20.7.2). Autovote allows for the creation of pre-filled-in paper ballots. Once again, this feature is enabled through Windows registry entries. Once these entries are enabled, Ballot Now displays the Autovote menu option when started.

The Autovote menu allows the Ballot Now user to choose the number of pre-filled-in ballots to print. The user has no control over the selected filled in entry for each contest, however, the selected entries are uniformly distributed. This allows an arbitrary number of ballots with the desired results to be printed with the overhead of some ballots with undesired results that may simply be discarded.

Paper ballots generated by Autovote initially say "Autovote" on the front and back, making them conspicuous and easy to detect in an audit or recount. We were able to overcome this by installing a PNG printer driver on the Ballot Now machine. This driver allows ballots to be printed to PNG image files as opposed to paper. We could then open the files in an image editor, remove the Autovote label and print them. Aside from the label, Autovote ballots are identical to regular ballots. We conducted a normal election and an election with Autovote ballots, and could not identify any differences in the eScan unofficial printout, the audit logs, or the cast vote records on the eScan's MBB.

Autovote could be used in tandem with the eScan's duplicate ballot feature to perform a ballot stuffing attack. Using Autovote ballots is advantageous over using photocopies, as each Autovote ballot has a unique serial number, and thus cannot be differentiated from legitimate votes in an audit.

Attack Class 20: Third-Party Software Vulnerabilities – The vulnerabilities listed throughout point to a general design failure: all components rely on third-party implementations for mission critical features. For example, the eCM tokens in use are Spyrus Rosetta USB devices with seemingly no validation of the cryptographic operations done by Hart. A Cryptoki API is exported that provides signing and encryption operations that are necessarily opaque; however, all of the trust in these tokens is reliant on the correct implementation of cryptographic functionality within these tokens, something that is difficult to validate when dealing with COTS hardware.

A potentially greater issue along these lines is the Hart system's extensive use of functionality from the underlying Windows operating system. In particular, the generation of eCM signing keys relies extensively on the `CryptGenRandom` function called by the Windows 2000 random number generator. Recent work by Dorrendorf et al. has shown that this generator contains vulnerabilities [19]. It is possible to find all previous states of the generator in about 19 seconds on a Pentium IV computer, and future keys may be predicted due to a lack of both forward and backward security in the PRNG. This vulnerability is outside of the scope of Hart's software to fix, meaning that there is a reliance on the willingness

of outside vendors to solve these sorts of potential vulnerabilities. The issue of potentially insecure back-end Windows systems has been discussed in previous reports on the Hart system, notably Issue 20 of the CA TTBR.

4 Conclusion

The results of the Project EVEREST study were significant - in less than nine weeks of study, the team discovered 27 new issues in the Hart system and doubled the number of publicly known weaknesses in Premier systems; given the increasing discovery rate at the close of the study, we expect many more issues remain. Moreover, not only were new vulnerabilities found, but also entirely new *classes* of vulnerabilities were found. This was most clear in the discovery of the vast number of undocumented features in the Hart system enabled by hidden Windows registry entries. Possibly more importantly, each previously unevaluated device we studied possessed significant security failures. For example, the ExpressPoll electronic poll book presented little or no challenge to compromise. In this case, the consequences of this are clear–anyone using this device could arbitrarily add voters or disenfranchise others.

Our analysis will certainly not be that last evaluation of electronic voting equipment. If and when the next study occurs, we expect that other researchers will find our methodology helpful. In particular, by forcing ourselves to begin with the confirmation of known vulnerabilities, we were able to quickly learn about the innerworkings of the Hart and Premier systems. This process not only added value to the community by providing independent validation of previously known problems, but also served to help us quickly identify new vulnerabilities in both previously evaluated and new components of the system. We recommend that future studies follow a similar model not only to create further confidence in the results of previous reports, but also to allow researchers in such studies to understand these systems as quickly as possible so as to allow them to identify additional serious weaknesses.

References

[1] Patrick McDaniel, Kevin Butler, William Enck, Harri Hursti, Stephen McLaughlin, Patrick Traynor, Matt Blaze, Adam Aviv, Pavol Cerny, Sandy Clark, Eric Cronin, Gaurav Shah, Micah Sherr, Giovanni Vigna, Richard Kemmerer, David Balzarotti, Greg Banks, Marco Cova, Viktoria Felmetsger, William Robertson, Fredrik Valeur, Joseph Lorenzo Hall, and Laura Quilter. EVEREST: Evaluation and Validation of Election-Related Equipment, Standards and Testing. `http://www.sos.state.oh.us/`, 2007.

[2] Tadayoshi Kohno, Adam Stubblefield, Aviel Rubin, and Dan Wallach. Analysis of an Electronic Voting System. In *Proceedings of the IEEE Symposium on Security and Privacy*, May 2004.

[3] A. Kiayias, L. Michel, A. Russell, and A. A. Shvartsman. Security Assessment of the Diebold Optical Scan Voting Terminal. UConn Voting Technology Research (VoTeR) Center, October 30, 2006.

[4] A. Kiayias, L. Michel, A. Russell, and A. A. Shvartsman. Integrity Vulnerabilities in the Diebold TSX Voting Terminal. UConn Voting Technology Research (VoTeR) Center, July 16, 2007.

[5] Elliot Proebstel, Sean Riddle, Francis Hsu, Justin Cummins, Freddie Oakley, Tom Stanionis, and Matt Bishop. An Analysis of the Hart Intercivic DAU eSlate. In *Proceedings of the USENIX/ACCURATE Electronic Voting Technology Workshop*, August 2007.

[6] Joseph Calandrino, Ariel Feldman, Alex Halderman, David Wagner, Harlan Yu, and William Zeller. Source code review of the Diebold voting system. University of California, Berkeley under contract to the California Secretary of State, July 20, 2007.

[7] Srinivas Inguva, Eric Rescorla, Hovav Shacham, and Dan Wallach. Source code review of the Hart InterCivic voting system. University of California, Berkeley under contract to the California Secretary of State, July 20, 2007.

[8] Robert Abbot, Mark Davis, Joseph Edmonds, Luke Florer, Elliot Proebstel, Brian Porter, Sujeet Shenoi, and Jacob Stauffer. UC Red Team Report: Diebold Elections Systems, Inc. University of California, Berkeley under contract to the California Secretary of State, July 2007.

[9] Robert Abbot, Mark Davis, Joseph Edmonds, Luke Florer, Elliot Proebstel, Brian Porter, Sujeet Shenoi, and Jacob Stauffer. UC Red Team Report: Hart InterCivic. University of California, Berkeley under contract to the California Secretary of State, July 2007.

[10] Ryan Gardner, Alec Yasinsac, Matt Bishop, Tadayoshi Kohno, Zachary Hartley, John Kerski, David Gainey, Ryan Walega, Evan Hollander, and Michael Gerke. Software Review and Security Analysis of the Diebold Voting Machine Software. Security and Assurance in Information Technology (SAIT) Laboratory, Florida State University, For the Florida Department of State, July 27, 2007.

[11] David Gainey, Michael Gerke, and Alec Yasinsac. Software Review and Security Analysis of the Diebold Voting Machine Software: Supplemental Report. Security and Assurance in Information Technology (SAIT) Laboratory, Florida State University, For the Florida Department of State, August 10, 2007.

[12] Fortify Source Code Analysis (SCA). `http://www.fortifysoftware.com/products/sca/`.

[13] D. van Heesch. Doxygen. `http://www.doxygen.org`.

[14] Roger G. Johnston. Tamper-indicating seals. *American Scientist*, 94:515–523, November-Decemeber 2006.

[15] Hart InterCivic. BOSS Operations Manual 6100-019. Rev. 43-62A.

[16] Ariel Feldman, J. Alex Halderman, and Edward Felten. Security Analysis of the Diebold AccuVote-TS Voting Machine. In *USENIX/ACCURATE Electronic Voting Technology Workshop (EVT)*, 2007.

[17] Microsoft Corporation. Microsoft Security Bulletin (MS00-0500). `http://www.microsoft.com/technet/security/Bulletin/MS00-050.mspx`, July 2000.

[18] Microsoft Corporation. Microsoft Security Bulletin (MS02-004). `http://www.microsoft.com/technet/security/Bulletin/MS02-004.mspx`, February 2004.

[19] Leo Dorrendorf, Zvi Gutterman, and Benny Pinkas. Cryptanalysis of the Random Number Generator of the Windows 2000 Operating System. In *Proceedings of the ACM Conference on Computer and Communications Security*, Alexandria, VA, November 2007.

Appendix A - Hart InterCivic Architecture

Here we briefly describe the Hart InterCivic Voting System by walking through a sample election procedure (as typical in Ohio); a more detailed description can be found in the EVEREST report [1]. Refer to Figure 3 for component orientation and interaction; all county headquarters components run on a Windows system.

Before the election begins, the eSlate Cryptographic Module Manager, or *eCM Manger* (5), is used to generate a cryptographic master key, which is stored on every eCM token (a Spyrus Rosetta USB cryptographic token) used in the election (i.e., there is one master key for a county). The Ballot Origination Software System, or *BOSS* (1) creates an election database, including precinct and race definitions and the corresponding ballots for every county precinct. BOSS then writes the data to PCMCIA storage cards called Mobile Ballot Boxes, or *MBBs* (7); one MBB is written for each Judge's Booth Controller, or *JBC* (8), and *eScan* (9) used in the county, along with one additional MBB to be used by *Ballot Now* (2) for recording absentee ballots. Meanwhile, in the warehouse, the System for Election Records and Verification of Operations, or *SERVO* (6), software is used to reset the memory of all JBCs and *eScans* (10) and to reset their vote count to zero. SERVO is also used to transfer the shared key from an eCM onto the JBCs and eScans.

On election day, voters using the eScan fill out paper ballots and enter them in the machine, which tallies the results and writes them to an MBB. Voters using the eSlate DRE first go to a poll worker, who provides them with a 4-digit access code generated by the JBC. The voter enters the code into the eSlate, which provides onscreen instructions for casting a ballot. The voter can verify their vote by checking the paper trail printed on the Verified Ballot Option (VBO) attached to the eSlate. The confirmed vote is recorded to the JBC's MBB and internal memory, and the internal memory of the eSlate. Absentee ballots are processed by Ballot Now, which records results to an MBB.

After the election, MBBs are retrieved and taken to election headquarters. In Ohio, these are either loaded directly to a machine running *Tally* (3), or onto a machine running *Rally* (4) that is on an internal private network with Tally. Tally tabulates results from each of the MBBs and produces an election result database along with a variety of reports. After election night, the audit logs and vote records from the JBC, eSlate, and eScan machines are backed up by SERVO and the firmware is verified.

Appendix B - Premier Architecture

In this appendix we briefly overview the Premier Voting System by walking through a sample election procedure (as typical in Ohio); a more detailed description can be found in the EVEREST report [1]. Refer to Figure 4 for component orientation and interaction. Note that our study is unique in its access to the EMP or ExpressPoll, as well as Verdasys Digital Guardian, a third party tool used to secure the GEMS server in Ohio counties.

Using the Global Election Management System server, or *GEMS* server (1), an administrator begins an election by defining a ballot. This includes determining the races, candidates and issues that will appear. When the ballot is approved, the GEMS server communicates over a local area network with either the *Central Office AV-TSX* (2) or the *Election Media Processor* (3), which encode 128 MB PCMCIA memory cards (7) used at the polling place AV-TSX. The Election Media Processor, or EMP, is a PC running either Windows 2000 or XP connected an external drive bay containing multiple memory card readers and is incorporated for efficiency reasons. GEMS also communicates with a *Central Office AV-OS Precinct Count* (4) in order to encode 128 KB EPSON 40-pin memory cards used by the polling place AV-OS. Memory cards are then sent to the polling station either independently or pre-inserted into voting machines, depending on policy. Also configured by GEMS at the county election headquarters is the *AV-OS Central Count* (5) (used for absentee ballots) connected via a *Digi PortServer II* (6), which multiplexes serial connections into Ethernet.

For counties using Premier touchscreen voting systems, a precinct administrator opens an election by inserting a *Supervisor Card* (a smart card) into the *AV-TSX* (8). After voters receive a *Voter Card* (9) from a poll worker with either the *Voter Card Encoder*, VCE (10) for short, or *ExpressPoll* (11) (an electronic replacement for the traditional voter log book, which runs Windows CE), they approach an AV-TSX and insert it into the machine. After casting their vote, the voter returns their used Voter Card and leaves the polling station. When the poll closes, the precinct administrator then reinserts the Supervisor Card and closes the election. Elections using optical scan units instead begin by having a precinct administrator place the device into election mode. Voters in these precincts fill out paper ballots and then feed them to the *AV-OS PC* (12), which scans their results. In both systems, memory cards are shipped back to the county elections headquarters at the close of elections for centralized tabulation.

Upon arriving at the county's election headquarters, memory cards are then inserted into the appropriate devices, which communicate the results of the election to the GEMS server over the local area network. The GEMS server then prints an official election results summary, which is used as the official outcome of the election.

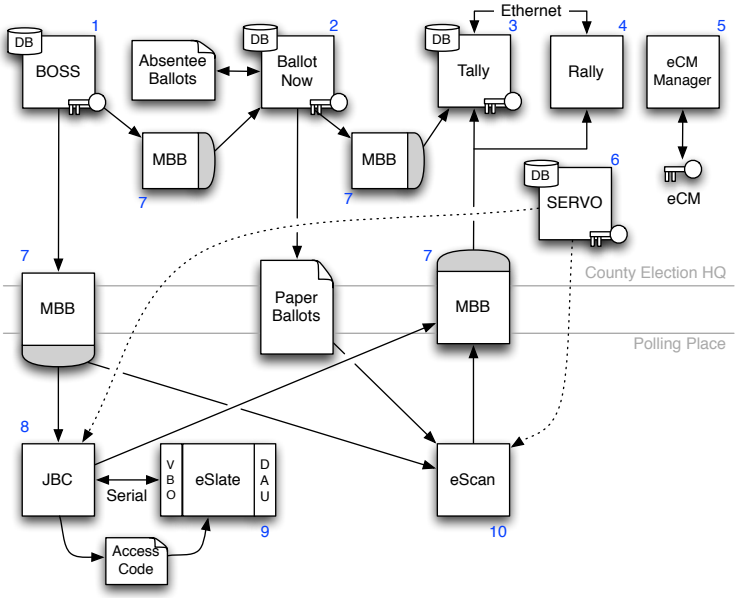

Figure 3: The major portions of the Hart system architecture and some of the connected components. Unless indicated otherwise, solid lines indicate a physical relationship between components, i.e., components are either physically connected or must be physically transported to interact with each other. Dashed light lines represent connections that take place between components outside the course of an operational election in progress.

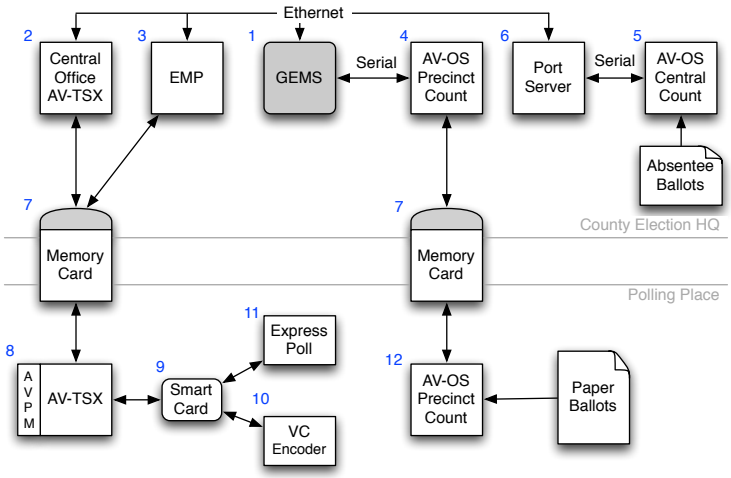

Figure 4: The major components and their relation to each other for counties using Premier Electronic Voting Systems. Unless otherwise stated, arrows depict physical transport of cards or ballots.