EVEREST: Evaluation and Validation of Election-Related Equipment, Standards and Testing[*†]

Final Report
December 7, 2007

[*] This report was prepared by teams from Pennsylvania State University, the University of Pennsylvania, and WebWise Security, Inc. as part of the EVEREST voting systems analysis project initiated by the Secretary of State of Ohio in the Winter of 2007. Unless otherwise indicated, all analyses detailed in this report were carried out at the home institutions between October 1, 2007 and December 7, 2007.

[†] This report is released by Ohio Secretary of State Jennifer Brunner consistent with the Ohio Public Records Act, Ohio R.C. 149.43. The reader of this document is advised that any conduct intended to interfere with any election, including tampering with, defacing, impairing the use of , destroying, or otherwise changing a ballot, voting machine, marking device, or piece of tabulating equipment, is inconsistent with Ohio law and may result in a felony conviction under, among other sections, Ohio R.C. 3599.24 and 3599.27.

Project Personnel

Pennsylvania State University Team

Patrick McDaniel, *Principal Investigator & Team Lead*

Kevin Butler
Pennsylvania State University

William Enck
Pennsylvania State University

Harri Hursti
Independent Contractor

Steve McLaughlin
Pennsylvania State University

Patrick Traynor
Pennsylvania State University

University of Pennsylvania Team

Matt Blaze, *Team Lead*

Adam Aviv
University of Pennsylvania

Pavol Černý
University of Pennsylvania

Sandy Clark
University of Pennsylvania

Eric Cronin
University of Pennsylvania

Gaurav Shah
University of Pennsylvania

Micah Sherr
University of Pennsylvania

WebWise Security, Inc.

Giovanni Vigna, *Team Lead*

Richard Kemmerer
WebWise Security, Inc.

Davide Balzarotti
WebWise Security, Inc.

Greg Banks
WebWise Security, Inc.

Marco Cova
WebWise Security, Inc.

Viktoria Felmetsger
WebWise Security, Inc.

William Robertson
WebWise Security, Inc.

Fredrik Valeur
WebWise Security, Inc.

Policy and Document Consultants

Joseph Lorenzo Hall
University of California, Berkeley

Laura Quilter
University of California, Berkeley

Part IV

Analysis of the Hart InterCivic, Inc. Voting Systems

CHAPTER 16

HART EXECUTIVE SUMMARY

This study evaluates the ability of the Hart voting system to conduct a trustworthy election. The review team was provided access to the Hart source code and election equipment. The reviewers studied these materials in order to identify any security issues that can be exploited to affect an election. As part of that analysis, the reviewers were asked to identify best practices that may limit or neutralize the impact of discovered issues.

Our evaluation suggests that the Hart system lacks the technical protections necessary to guarantee a trustworthy election under operational conditions. The vulnerabilities and features of the system work in concert to provide numerous opportunities to manipulate election outcomes or cast doubt on legitimate election activities. Such vulnerabilities are exploitable under election conditions, and often require minimal physical access to equipment or information. These vulnerabilities are a result of the following failures of the Hart system's design, implementation, and practices:

- *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 even brief access to the voting systems. These vulnerabilities place enormous burdens on physical procedures.
- *Failure to eliminate or document unsafe functionality* - There are a number of largely undocumented features in the system that are highly dangerous in a production election system. For example, existing features allow an attacker to remotely "script" DRE voting machines to cast votes as the attacker chooses, to allow a single (or photocopied) voter ballot to be counted many times, and to print pre-voted ballots that will be accepted by voting equipment. Note that all of these activities are not attacks *per se*, but are the apparent intended use of existing Hart system features. These features are available during a live election.
- *Failure to protect election from malicious insiders* - The protections in the Hart system that are intended to prevent election officials, poll workers, and vendor representatives from using dangerous features or modifying election data are circumventable. Attackers with access to the system can quickly recover critical system passwords, extract cryptographic keys, and reproduce security hardware. These artifacts are the "keys to the kingdom" that can be used to forge election data and compromise nearly all of the Hart election equipment.
- *Failure to provide trustworthy auditing* - The auditing capabilities of the Hart system are limited. Those features that are provided are vulnerable 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.

One of the critical discoveries in this study is that the full functionality of the Hart system is currently unknown, and in some cases may never be known. We found many functions and system configurations that were not documented and whose purpose was non-obvious. The vast majority of these remain unstudied for lack of reviewer time. Furthermore, certain interfaces–in particular in the election tally software–were designed to be augmented at run time with additional software. Because these interfaces were apparently designed to allow previously unknown software to be introduced into the live system, they represent a source of potential vulnerability.

Our findings are consistent with those of previous studies. The lack of protections leave the system vulnerable. Thus, the security of an election is almost entirely reliant on the physical practices. The technical limitations of its design further show that when those practices are not uniformly followed, it will be difficult to determine if attacks happened and what they were. Even when such attacks are identified, it is unlikely that the resulting damage can be contained and the public's confidence in the accuracy and fairness of the election restored.

CHAPTER 17

HART STUDY OVERVIEW

17.1 Part Structure

The following chapters detail the Hart portion of the EVEREST review. Readers are directed to Part 1 of this report for a discussion of the review's goals and methodologies. Those readers not experienced in information security or Ohio election practices are also encouraged to read the threat model in Chapter 3. The content in that chapter is instrumental in gaining a detailed understanding of the substance and impact of the issues identified throughout.

The issues and commentary described in this section of the report are *reflective of our analysis of the Hart system only*. None of the included material should be considered to apply to either the ES&S or Premier systems, nor should any statement be construed to be making any quantitative or qualitative comparisons of the different systems. Such comparisons are expressly outside the scope of the review, and we have not developed any opinions on the relative merits of any one system with respect to the others. Nothing in this review should be considered as a positive or negative commentary on electronic voting in general.

The remainder of this part of this report is structured as follows. We begin in the next section by giving a brief overview of the Hart system and its use in Ohio elections. Chapter 18 broadly characterizes the security and reliability of the Hart system by highlighting several architectural and systemic issues encountered in the review. Chapter 19 provides detailed technical information on the confirmation of previously reported issues and Chapter 20 provides similar detail on new issues uncovered by our analysis. Chapter 21 describes a number of attack scenarios that demonstrate how the identified issues may be used in concert to subvert an election.

17.2 Architecture

This chapter provides an overview of the architecture for the Hart InterCivic Voting System, including the Election Management System (EMS), the devices comprising a voting setup, and some of the back-end software in use. This information is compiled from public reports and experience of the Ohio EVEREST academic team with the system.

Figure 17.1 provides an overview of the setup and interconnected components within the Hart InterCivic voting system. The Hart system is comprised of a number of components, and can be delineated mainly between components found at county election headquarters versus those found in polling locales.

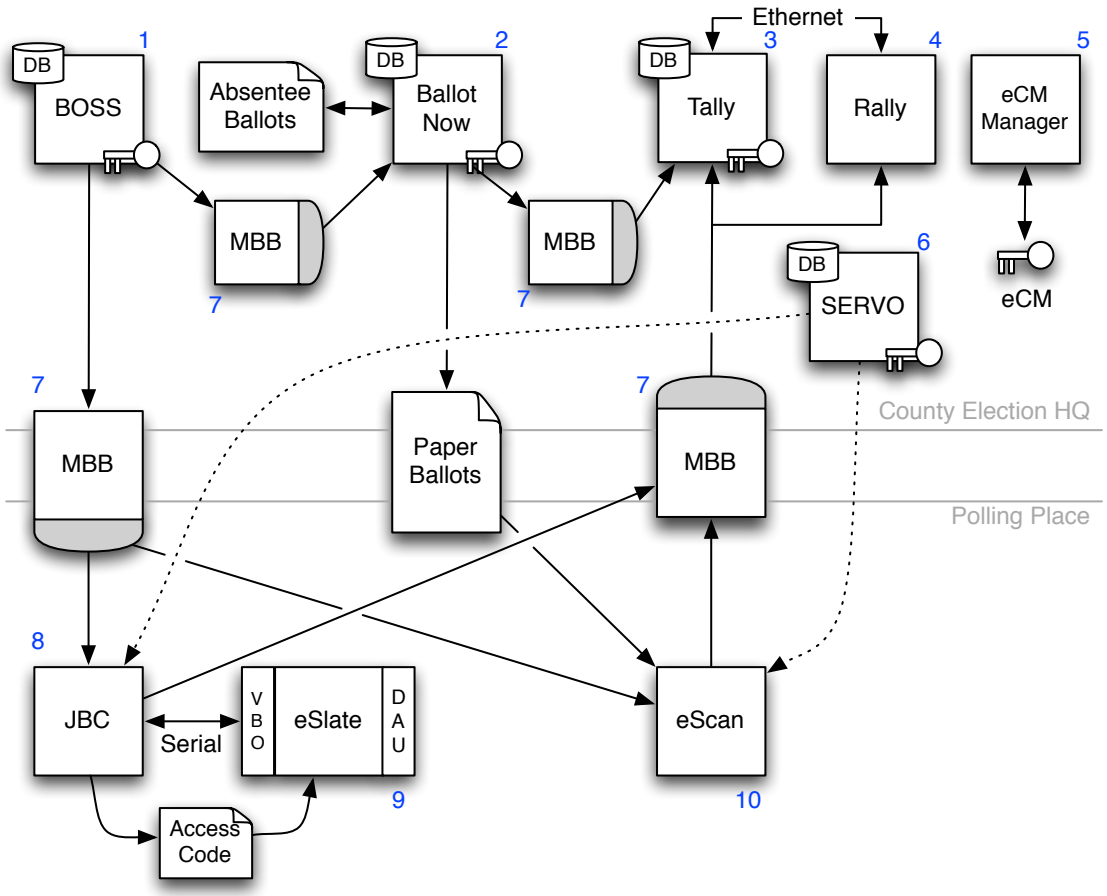

Figure 17.1: The major portions of the Hart system architecture and some of the connected components. Unless indicated otherwise, solid lines indicate a physical relationship between components, i.e., components are either physically connected or must be physically transported to interact with each other. Dashed light lines represent connections that take place between components outside the course of an operational election in progress.

17.2.1 Components at County Election Headquarters

A number of components within the Hart system reside at the county election headquarters. Refer to Figure 17.1 for a pictorial representation of component interaction.

1. **BOSS:** The Ballot Origination Software System is used to set up an election. It creates an election database (i.e., lists of candidates running for each office, precincts, etc.) and is used to define ballot templates, with information sourced from a Sybase database. In addition, passwords for the polling place voting equipment (the JBC and eScan) are defined in BOSS. BOSS exports election definitions to Mobile Ballot Boxes (MBBs). BOSS runs on a Windows 2000 server, and apart from the MBBs it generates, is disconnected from the rest of the election process.

2. **Ballot Now:** This software is used to generate paper and electronic ballots for elections. An MBB is created for Ballot Now by BOSS. With the information on this MBB, Ballot Now generates ballot

images for all districts, creating ballot images electronically for use in the eSlate machines and in a printer-ready form for use in eScan machines and for absentee ballots. In addition, it may be used with a commercial central count optical scanner to process absentee ballots, writing out an MBB containing all vote totals. Ballot Now runs on a server running the Windows 2000 Server operating system.

In reality, Ballot Now is a client-server architecture. The server attaches to the scanner used for counting absentee ballots. The clients are used for manual control of this counting process. We did not have the scanner, nor the networking elements to evaluate. Thus, there may be any number of additional issues that we could not identify for lack of equipment.

3. **Tally:** Tally accumulates and generates a tally of all votes for the electoral district. It runs on a Windows 2000 Server machine and is used to compute total votes based on the MBBs loaded into it from the eScan and JBC devices, and from Ballot Now. The database defining the election in BOSS is imported to Tally and used to provide candidate information and other information, including how many MBBs were generated and how many should be read. Once all MBBs have been tallied, Tally can be used to generate reports about the election outcome. Tally may also be used to manually adjust vote totals. MBBs that are not loaded directly into Tally are retrieved remotely from *Rally* (described below).

4. **Rally:** Rally is used to process MBBs from precincts. Its main purpose is to generate vote totals to communicate to Tally. In Ohio, Rally must be used at election headquarters and may only communicate with Tally over an internal private computer network; however, in other jurisdictions, Rally may be used remotely and communicates with Tally over a private network or modem connection with Tally, rather than physically transporting MBBs to the central locale. An SSL connection is established between Rally and Tally to secure the communication between the two processes. It runs on a Windows 2000 Server machine.

5. **eCM Manager:** This software is responsible for generating signing keys for eSlate Cryptographic Module (eCM) tokens. An eCM is a Spyrus Rosetta USB cryptographic token that stores a key. The token must be inserted for certain sensitive functions within BOSS, Ballot Now, Rally, Tally, and SERVO to operate. eCM Manager writes a key identifier and the generated signing key to each eCM token, and sets up a 6-digit personal identification number (PIN) to access the token. eCM Manager operates on a PC running the Windows 2000 Server operating system.

6. **SERVO:** The System for Election Records and Verification of Operations tracks inventory for all election devices used within a county; all serial numbers for devices are stored in this application, as is the firmware version associated with the device. In addition, SERVO installs cryptographic keys on JBC and eScan devices and resets their memory prior to the beginning of an election, backing up the memory, CVRs, and audit logs for these devices to a database. SERVO operates on a PC running Windows 2000 Server and connects to an eScan over Ethernet, and to the JBC with a parallel cable.

17.2.2 Components in Use at Polling Locations

The following components make up the Hart precinct voting system. Please refer to Figure 17.1 for a pictorial representation for how the components interact.

7. **MBB:** A Mobile Ballot Box is a PCMCIA card that stores information such as ballot definitions and a set of cast ballots. It is the primary means of transporting election information between components

within the Hart system. Ballot definitions are transported through an MBB from BOSS to Ballot Now, where it is used to generate ballots. MBBs generated from BOSS are also used in the JBC and eScan devices for collecting vote information, which are subsequently tallied using the Tally server.

8. **JBC:** The Judge's Booth Controller is a console that can control up to 12 eSlate DREs attached to it. The JBC generates voter access codes and distributes ballot configurations to the eSlates; it also records the Cast Vote Records (CVRs) and stores eSlate ballots to internal memory. In addition, ballots are written to an MBB. The JBC is capable of accumulating and reporting vote results. It connects the eSlates in a dumb-terminal fashion with an RS-485 cable connecting the HD15 interface on the eSlate to the DB9 port on the JBC.

9. **eSlate:** This is the direct-recording electronic (DRE) voting unit used in a Hart-run precinct. It attaches to a *Verified Ballot Option* (VBO) printer known as the *VBOx* to produce a voter-verified paper audit trail (VVPAT). The eSlate may also contain a Disability Access Unit (DAU) with magnified text rendering for vision-impaired voters and oversized "jelly switches" for voters with tactile impairments. In all units, instructions and confirmations are given through an audio track as well as visually on screen.

10. **eScan:** This is a precinct-based optical ballot scanner for voters who choose to vote using paper ballots (currently all non-disabled voters in Ohio). The eScan scans and tabulates votes based on the ballots deposited into it by the voters, and contains an MBB used to store tabulated vote results.

17.3 General Election Procedure

Here are the general steps used to conduct an election using the Hart system:

17.3.1 Pre-Election Procedures

- The eCM manager generates a cryptographic master key, and stores a copy of this key on each eCM to be used during the election. A PIN is required to store the key.

- BOSS creates an election database based on data sourced from the back-end Sybase database. This includes precinct and race definitions, ballot definitions, and other options, such as party primaries, to create the right number of ballot variants. An eCM token must be plugged into the machine where BOSS runs for the generated ballots to be accepted.

- BOSS is used to "burn" or write MBBs. The number burned is dependent on the county size and the number of JBCs and eScans in the area. An audio card (also a PCMCIA card) for each eSlate with a DAU is also burned by BOSS at this time. An MBB is burned for specific use in Ballot Now; it cannot subsequently be used in any other component or an error message will result. BOSS tracks the number of burned MBBs. Once all MBBs have been burned, the election may be finalized. After this time, no more MBBs can be burned, and no election configurations can be changed.

- Ballot Now takes an MBB generated by BOSS and generates electronic images of the ballots. These can be printed with a standard laser printer at county election headquarters or the images may be sent to commercial printers for ballot generation.

- In the warehouse where voting equipment is kept, SERVO is used to reset the memory of all JBCs and eScans and to reset the vote count to zero. When an eCM token is plugged into the machine running SERVO, the cryptographic master key for the country may be installed into the JBCs and eScan devices.
- MBBs are installed into the JBCs and eScans and are now ready for polling. The devices are tamper-sealed at the warehouse. The MBBs may also be transported to the polling locations to be installed into the eScans and JBCs there.

17.3.2 Election Day Procedures

- Election administrators put passwords into JBCs and eScans to start them. A separate password is used to open the polls, after a zero-tape is printed from each device.
- During an election, voters using eScan machines who have registered receive paper ballots, to be filled out with blue or black ink. These ballots are placed in the eScan machine. If the ballot is over-voted (i.e., two candidates are chosen in a race where only one is allowed), the ballot will be returned to the voter and invalidated. The voter can then fill out and submit a new ballot.
- In Ohio, disabled voters are able to use eSlate DRE machines. When they have registered, they will proceed to the poll workers staffing the JBC management station. Each voter will receive a 4-digit code. The voter proceeds to the eSlate where they enter the code and choose candidates, using the on-screen instructions for guidance. eSlate machines with a DAU installed will provide audio narration for the voter. A VBO printer is attached to the eSlate, which provides a paper confirmation of the voter's choices. When the voter casts their ballot, the paper is advanced out of their view. A voter has three chances in Ohio to cast their ballot in case they cancel their ballot choices.
- At the close of the election, election administrators enter passwords into JBC and eScan machines to close the polls.

17.3.3 Post-Election Procedures

- After the election, MBBs from the JBC and eScan units are either physically taken to central headquarters where they are either directly loaded into Tally or loaded into a machine running Rally, which is set up in Ohio to communicate with Tally over an internal private network. (Rally requires an eCM to be inserted for MBBs to be read.)
- Paper ballots, such as absentee ballots [1] precinct, are transported to a central facility and read by a specified central count optical scanner, with the image recognition done by Ballot Now. The results are written to an MBB and sent to Tally for tabulation.
- An eCM token is inserted into the machine where Tally runs, which upon receiving all of the MBBs, tabulates the votes, using the token to ensure no tampering took place with the results. Tally produces an election result database and a variety of reports.
- SERVO backs up the CVRs and audit logs from the JBC, eScan, and eSlate units. These can be used to produce recount MBBs to be recalculated in Tally if necessary. The firmware of all devices can also be verified by SERVO to ensure it is the same as when the machines were sent out.

[1] It also is possible that paper ballots may be used in a precinct without being processed by an eScan. In this case, the ballots will be transported to a central facility and processed by Ballot Now.

17.4 Data Security

The central mechanism used in the Hart system to provide data security is a single county-wide cryptographic key (a secret value, similar to a password, shared throughout the county). This key is generated by eCM Manager and is later programmed into eCM hardware tokens (similar to USB "thumb drives") and all precinct devices, prior to sending them to the field. The back end systems (BOSS, Ballot Now, Tally, and Rally) require an eCM token be present before allowing access to critical functions. The key is read from the token and into the computer's main memory.

During the preparation for an election, the key is read by SERVO and placed in the memory of the eScan and JBC during the "reset" operation (in preparation for an election). BOSS prepares MBBs for use in an election by pushing ballot data onto the memory card and digitally signing[2] it. The MBBs are either sent to the precints on the day of the election, or placed in the JBCs and eScans for use in the election and shipped to the local precincts—sometimes being stored in insecure locations overnight.

The digital signatures of the MBBs are checked at the time that the parent JBC or eScan is booted (typically the morning of the election), and rejected if they are incorrect. The polls are opened and the MBBs begin receiving voting and audit data. When the polls are closed, the MBB is signed using a copy of the key in the parent device's memory. The JBC and eScan keep counters of the number of votes cast in internal flash memory. One of these counters is maintained over "reset" operations.

The MBBs and associated VVPATs are returned to the county election headquarters after the polls have closed. In Ohio, Tally is used to tally all of the votes on all of the MBBs whose digital signatures are valid. Recounts are mandated by statute in extremely close races. The tabulated votes are inserted into a database on the Tally host. Election result reports are generated from the Tally database. 11 to 15 days after the election, a canvass is held to validate and verify the results. [3]

17.4.1 Other Components in the Hart System

Hart has written a number of additional utilities that may be used in a jurisdiction. They include the following:

- **Fusion:** This utility is used to consolidate multiple sources of tallied report information. Descriptions from the Hart website state that "The Fusion software application is a flexible tool that provides election results integration and enhanced reporting options. Fusion can be used to combine election results from different Tally databases, or to combine Hart Voting System election results with results from third-party systems. In addition, Fusion can import results from election databases to provide enhanced reporting options." [4] It has not been certified for use in Ohio, but appears to be used in some jurisdictions. We were not given access to either the application or its source code.

- **InFusion:** This utility is used to retrieve information from multiple sources when creating ballot definitions. It is not certified for use in Ohio and we are unclear as to whether it is used in any jurisdictions. We were not given access to either the application or its source code.

- A utility called **Bravo** exists but we do not have any information about it. The only reference to it is

[2] In reality, this is not technically a digital signature, but a hashed message authentication code.

[3] See ORC Ann 3505.32 for more information on the canvassing procedure in Ohio.

[4] `http://www.hartinc.com/pages/171`

in the *eSlate Reference Manual* but it is not described there nor anywhere else. This software is not certified for use in Ohio and we are unaware of whether it is used in any jurisdictions.

- The Hart JBC is able to connect with **Voter Registration** systems, but there is no information about what these may be, whether there is a system used by Hart, or any other information. No Hart-branded voter registration systems are certified for use in Ohio and we are not aware of any jurisdictions where any of these systems may be used.

CHAPTER 18

HART SYSTEMIC AND ARCHITECTURAL ISSUES

This report focuses on the ability of the studied system to conduct an election without fear of manipulation of its outcomes, processes, or public perceptions. As stated in the executive summary, we have found substantial flaws that inhibit the ability of the Hart system to meet these goals. Many of the issues presented in the following chapters relate to functional or procedural problems embodied in the Hart design and implementation. Such problems undermine the security of the system and provide attacker-accessible avenues to critical election data and systems. This chapter broadly characterizes the report's findings by highlighting the four areas of most concern:[1]

- **Ineffective data, software, firmware, and communication protections** - the majority of security mechanisms used to ensure data and software integrity can be bypassed by an attacker.
- **Ineffective authentication** - all user and system authentication methods (passwords, hardware tokens, cryptographic keys, etc.) can be bypassed or otherwise subverted, often trivially, by an attacker.
- **Undocumented, unprotected, and unsafe features** - the Hart system contains many unsafe and frequently undocumented features that are enabled through simple and available configuration interfaces.
- **Design, development, and maintenance Issues** - The pervasive lack of sound software and security engineering practices is the source of ubiquitous security, reliability, and maintenance issues.

Our findings are fully consistent with previous studies. In particular, we found the architectural and systemic findings of the Hart Source Code Report of the California TTBR study to be universally true. Here, we extend those assertions with our own, and attempt to more generally characterize the specific classes of security issues within the Hart system.

Note that the descriptions below are simply a sampling of the problems that arise from confirmed and newly identified issues. There are many other attacks resulting from these and other areas that can have serious negative consequences for perceived and real election validity.

[1] **Style note**: This chapter references numerous issues raised in the following chapters as demonstrative or enabling of some broader concern. These issues are denoted by reference to the chapter/section in which they are defined. For example, issue 19.2.1 discusses the use of a county-wide eCM key, and is be referenced as "(19.2.1)". Readers are directed to the referenced section for in-depth detail of the issue, its impact on the system, and procedural or technical mitigations, if any exist.

18.1 Ineffective Data, Software, and Firmware Protections

The mechanisms which Hart uses to protect data and software are frequently based on absent or flawed security models. The failure of these mechanisms to address important threats results in broad exposures of election data and features to an attacker. Such failures are a result of unrecognized requirements and other design and development issues. In most cases these issues cannot be addressed via software upgrades, but call for rethinking of both technical designs and procedural practices. We detail some of these central protection issues below.

18.1.1 Data

Much of data security in the Hart system flows from the single 32-byte key that is distributed throughout a county (19.2.1). This is poor security design–compromise of any one of potentially hundreds of devices is sufficient to subvert an entire county. Here the Hart design violates a basic *isolation* tenet of security engineering: compromise of a single precinct provides materials to compromise any precinct and the election headquarters. Further, if such compromise occurs, it will be impossible to identify which precinct is responsible for the breach.

In truth, obtaining the county key is trivial to an attacker with only a few seconds of physical access to precinct or back-end equipment–they can download the key from the eCM manager into a file (19.2.2) or extract the key from an eScan's memory via the unprotected Ethernet port (19.1.7), or by other means.

Once an attacker has the county key, they can forge any election data they want (19.1.7)–such modification would only be detected by careful comparison with the relevant VVPATs or physical ballots. However, as the VVPAT is also forgeable (20.5.6), this countermeasure is only of limited use. If both the MBB and VVPAT/completed ballots are forged, then there will be no way of detecting the forgery short of studying the internal audit information in each eScan and JBC used. This audit information itself can be erased (20.4.2) by an attacker with physical access to the device. These issues can lead to undetectable precinct-level forgery of an election (see Chapter 21.3).

The integrity and secrecy of election and audit data is protected on the back-end EMS servers in databases. Sometimes this data is stored in encrypted format, and sometimes it is not (19.1.3). Further, the passwords for the EMS applications and the databases they use are available (19.1.2) and easily bypassed (20.1.3). Once database access is gained, the attacker can modify any data they want, potentially manipulating election results or audit data. From an information security standpoint, these issues render the data held in server databases as functionally unprotected.

18.1.2 Communication

Hart is a networked system. The back-end and polling place devices speak to each other over serial, parallel, Ethernet, and other computer-to-computer interfaces. Such communication is inherently dangerous, as an attacker can masquerade as a remote party (e.g., pretend to be SERVO to a JBC) or "listen and modify" to the communications passing between the devices (e.g., modify commands passing between a JBC and an eSlate). It is an obvious and *essential* requirement that device-to-device communication be protected.

With limited exception, Hart provides no device-to-device communication security. This exposes critical data and functionality to the attacker. For example, and attacker may generate voter codes (20.4.1), upload

firmware (19.4.1), and erase voting or audit data (20.4.2). Where Hart does provide secure communications–Tally to Rally communications (over an internal network as mandated in Ohio) and for communication between Ballot Now and a security database–it uses a flawed protocol implementation (19.9.5).

18.1.3 Software and Firmware

The Hart software and firmware internal validity checks, where present, are ineffective at detecting compromise, either because they are designed such that they can never fail (19.7.1), or they can be trivially spoofed (19.7.2, 19.5.4). In the case of the eScan, the entire firmware can be replaced by an attacker with unobserved access to the eScan for 60 seconds (20.3.1). This latter attack would allow an attacker to completely alter the election results recorded on the MBB.

18.2 Ineffective Authentication

Authentication verifies a user, computer, or program's identity. Such identification is needed to ensure that only authorized parties perform security sensitive operations, e.g., create a MBB. We have found that every authentication mechanism in the Hart system is circumventable. The authentication methods include:

- **Hardware tokens** - Spyrus Rosetta hardware tokens store the county-wide keys, and are used in part to authenticate users of the Tally, BOSS, Ballot Now, eCM Manager, and SERVO applications. Tokens must be inserted into the Hart server before certain critical functions are enabled, e.g., the creation of new MBBs. These tokens can be created by anyone with access to the county-wide key (19.2.1). The Spyrus tokens are commercially available, and thus attackers can purchase them and create forged tokens by installing the key to them. The keys can be obtained either via download from eCM Manager (19.2.2), read directly from an eScan (19.1.7), or extracted from another token (20.2.1).

- **Passwords** - All Hart back-end EMS applications request a password when they are started. The cryptographic hashes of these passwords are stored in security databases, which are themselves easy to read and modify (19.1.3, 19.1.4). These databases are protected by passwords which are obfuscated using a trivially reversible technique (19.1.2). So, an adversary with access to the computer can read or create passwords to gain access to the server applications. Further, the applications all implement a flawed password policy that permits the user access if no password data is found in a security database (20.1.3) (after prompting a user for a new user/password).

- **JBC/eScan Access Codes** - MBBs contain the access codes used to enable administrator operations such as opening and closing the polls on the eScan and JBC. The MBB is not encrypted, so the codes can easily be read by an attacker (19.1.7).

- **Voter Codes** - Voter codes are used to authenticate voters before they can vote on an eSlate. The JBC generates the first voter code at random, then repeatedly uses a simple mathematical function on that code to generate each subsequent voter code. This process is perfectly predictable; an attacker who knows any voter code can determine all voter codes that follow it (and preceded it) in the order in which they will be assigned (19.7.6).

18.3 Undocumented, Unprotected, and Unsafe Features

The Hart system consists of many thousands of lines of code distributed over a large number of applications. The code was developed over nearly a decade by an apparently revolving staff of software engineers, based on the number of different authors and styles of coding encountered. A byproduct of this process is that there is a large number of old, unused, and otherwise "orphaned" features built into the software. Such features are often a source of security issues–they can be enabled in subtle ways. We review several of the more potentially dangerous of these functions below.

Note that there are likely many features in the Hart system that the review team was not able to identify–given time constraints, we were only able to understand a tiny fraction of the source code. Thus, we recommend that future development efforts exhaustively audit the code for such features.

18.3.1 Autovote

The Hart system provides an "Autovote" feature as part of the Ballot Now back-end server application (20.7.2). This feature allows election officials to print in bulk eScan ballots whose votes are pre-cast (or alternately that are not pre-cast). A report describing an expected tally of the printed ballots is also provided. The ballots are only accepted when the eScan to which these Autovote ballots is fed are in "test" mode. Each ballot has the words "AUTOVOTE" printed on the side making it visually distinct from others. Autovote is not enabled by default, but is only available when a combination of registry entries is set to specific values.

Autovote is not mentioned in the current product manuals or other provided technical documentation. It was briefly mentioned in a manual from 2001 (this reference was subsequently removed) and discussed in several hearings on voting systems in 2001 and 2003.[2]

Other security issues arise when a specific registry entry is set. When set, the ballots that are created will be accepted in the "election" mode—thereby allowing the ballots to be counted in a live election. Furthermore, if the registry entry is set when using Ballot Now to process ballots from a high speed scanner, Autovote ballots created without the override being set will be accepted as legitimate. Obviously, this is a very dangerous feature–an attacker (whether insider or outsider) who has access to the Ballot Now server and a precinct scanner could fraudulently generate vote tallies that would be accepted in a real election. It also may be possible to change the "AUTOVOTE" banner to some other text, including nothing, so that Autovote ballots would not visibly appear distinct from legitimate ballots.

Autovote is obviously test apparatus. Performing large-scale election testing requires the generation of ballots to scan. However, its inclusion in a production product is extremely dangerous.

18.3.2 Windows Registry Misuse

The Windows registry is a operating system service that maintains configuration parameters for applications installed on the computer. The Hart system makes extensive use of the registry to enable/disable features of the system. While in general the use of the registry is not a problem, Hart uses it to enable critical functions and security sensitive operations. Issues arise because anyone with the appropriate privileges on the computer can read and change the registry. Thus an attacker without any Hart system passwords or

[2] State of California, *Meeting: Secretary of State Voting Systems Panel*. December 16, 2003 ⟨URL: `http://www.sos.ca.gov/elections/vsp_min_121603.pdf`⟩.

hardware tokens can affect the security and behavior of the system. Examples of registry misuse include enabling Autovote for live elections (20.7.2), defining completely the available menus in Tally (20.6.2), and storage of the TLS password key used by Ballot Now (20.1.2).

An interesting characteristic of the registry use in the Hart software is that it (generally) periodically checks registry entries, rather that just checking them at start-up. This has the consequence that triggered features can be turned on and off without restarting the software.

We found references to many dozens or more of registry entries used by the Hart EMS applications. We were only able to investigate a small number of these. The vast majority of registry entries are undocumented, and their purpose is often unclear. An inventory of these items and their function should be constructed and documented for future certification and evaluation activities.

18.3.3 Remote eScan access

There is a server that allows users to remotely login to the eScan device (20.3.2) via the network connector (an unsealed RJ45 Ethernet port on the back of the eScan). This login capability is available before, during, and after an election. The eScan allows remote access to a local Windows `telnet` server. The Windows `telnet` has a long and checkered history of buffer overflows and other vulnerabilities. This represents a attractive target for an attacker to gain access to and compromise the eScan device. It is unclear what the remote access function is intended to be used for.

18.3.4 Adjust Vote Totals

The Tally application allows an attacker to directly change vote tallies through an edit screen available from the application menu (19.9.1). An entry is noted in the audit log. Subsequent election result reports provide no notification that the tallies were modified. This obviously represents a conduit for misuse by an attacker–particularly for a malicious insider. This appears not to be an orphaned function, but was intentionally included to enable local officials to correct election errors.

An attacker who misuses the adjusting vote totals interfaces can cover his tracks by forwarding the election reports, then restoring a previously un-adjusted database. Note that the frequent backing up and restoring of database is a common and vendor recommended operation. The restoration of the database effectively erases the adjustment from the local audit logs. The attacker can also modify the audit log directly (20.1.4).

18.4 System Design, Development, and Maintenance Issues

The lack of sound software and security engineering practices leads to pervasive quality and security problems. We briefly highlight some of the engineering quality problems encountered during the review.

- **Inadequate Low level Design** - The Hart system did not consistently apply an observable design methodology. Standard top-down abstraction and modularization practices were not followed, leading to convoluted, excessively complex, and sometimes circular flows of control within a program. The mixed use of programming languages, while common in commodity software, was improperly managed. This further complicated the design where the use of objects in C++ and C functions was

uneven, functionality was often duplicated, and relationships between data and user interfaces were not often clear or consistent.

- **Inadequate Coding** - The coding errors were universal. Buffer overflows, `printf` errors, and other violations pervade the code (19.1.12, 19.3.1). Note that these problems are often subtle and very difficult to identify after the fact. Excising these errors from a large system is an enormously time-consuming and difficult task. Standards for source code safety should be established and documented, and the source code should be audited for compliance.
- **Maintenance** - The uncontrolled evolution of the system is evident from the code. Features were created and abandoned, interfaces were changed in some parts of the system but not others, and many "hacks" (code fixes that quickly solve a problem, but do so in a suboptimal way) were added to the code. Such modifications accelerate the aging of the system and can incrementally make the system less stable and secure.

The lack of sound practices will seriously undermine any attempt to definitively verify the security of current and future versions of the software. The issues introduced by poor practices are not likely to be encountered in simple testing procedures, as they are typically found only by carefully crafted inputs and environments. Such inputs and environments are only identified through careful and lengthy analysis. However, the verification process is heavily stacked against the reviewers; they need to find all possible vulnerabilities (of which there maybe hundreds or more), whereas the attacker only needs to find one that the validation engineers didn't.

18.5 Audit Procedures

A fundamental and critical requirement for a complex system such as election management is the ability to audit every element of the system. Audit logs serve a vital purpose, as they can alert an auditor of suspicious or uncommon events that occurred, which could indicate the presence of malicious intent against the system. Because of this, it is critically important that an auditor may be completely confident that information retrieved from audit logs is complete and accurate. If an attacker can modify or destroy an audit log, it has no value. Without auditing, it is impossible to be confident that the outcome of an election is legitimate.

All of the voting equipment used in the Hart InterCivic System keeps an audit log. These are stored in different ways depending on the equipment in question. With the exception of the eCM Manager application, all of the back-end EMS software applications (i.e., BOSS, Ballot Now, Rally, Tally, and SERVO) have their own audit logs, stored in a database file associated with each application. During an election, the MBBs inserted into eScans and JBCs, and those written by Ballot Now as it processes absentee ballots, will store copies of audit logs. As a measure of redundancy, the eScan optical scanner stores an audit log inside its internal memory. In addition, the JBC and any eSlates connected to it will all store a copy of the audit log in their internal memory. Because of the VVPAT requirement in Ohio, a paper record is kept of all eSlate votes to be used in case of a recount, while the paper ballots processed by the eScan provide a physical record of cast votes. Note that the VVPAT log can be disabled via software.

All of these logs are subject to manipulation and deletion. The logs of all software applications can be modified by accessing the databases in a simple fashion (20.1.4). The JBC and all associated eSlates can have their audit logs erased from their internal memory by a malicious voter with a handheld computer (20.4.2) and eScan machines are similarly vulnerable (20.3.7). The audit logs on the MBBs in these machines may

be cleared at the same time the internal memory logs are cleared. The VBO printer attached to eSlate machines can be easily opened and the paper record removed (20.5.4). Finally, the paper ballots processed by the eScan can be damaged (20.3.5) and the integrity of the count may be compromised by stuffing the unit with duplicate ballots (20.3.6).

Taken in isolation, each of these attacks may seriously affect the auditability and ultimately, confidence in the election process. With just a small number of well-placed insiders, however, or a combination of insiders and malicious outsiders, it is possible to compromise logs at the polling places and at election headquarters, resulting in a catastrophic loss of verification and accountability for the county. As every piece of the system is vulnerable to attacks against audit logs, there are insufficient protections within the Hart voting equipment and software to prevent a motivated adversary from compromising an entire election.

CHAPTER 19

HART ISSUE CONFIRMATION

The sections of this chapter detail our confirmation of the issues raised in the Hart Source Code report developed as part of the California TTBR study. The sections are organized by component, and provide a brief summary of each issue raised, detail of the substance and potential exploitation of the issue, the means by which the issue was verified, and a strategy for the mitigation of the issue (if any level of mitigation is possible). The public version of this report contains references to sections of the private report. These redacted sections contain confidential information identifying where in the source code or design the issue is manifest, or other vendor-confidential information. Note that all source code, file names, and other apparent vendor specific information in the public version have appeared in previous election study analysis reports.

We were able to confirm 35 of the 36 issues raised in the original California TTBR report. The only exception was issue 14. In that analysis, we were able to identify a dangerous software function (`crc16`), but could not find a means of exploiting its design to attack the system in the short period we allotted to its investigation. However, because we could not view the private version of the California report, we did not have a means of evaluating the specific issue uncovered by the California reviewers. The Hart code is very complex, so it seems likely that the issue exists in the code, but the lack of usable detail prevented us from finding it.

19.1 Hart General Vulnerabilities

19.1.1 Corrupt MBBs can cause Tally to crash

The original California TTBR report suggests that Tally could be crashed by manipulating the data in the MBB in a specific way. The idea was to indicate via internal data that the card was larger than it was in reality. Tally would then process the MBB as if it were larger than it was, and crash as a result.

Our evaluation *cannot conclusively confirm or deny the issue*. We did review an exceptionally unsafe module in the source code that could be abused if the wrong information was sent to it. However, we could not immediately find a way to exploit this code. Because we could not view the private version of the California report, we did not have a means of evaluating the specific issue uncovered by the California reviewers. The Hart code is very complex, so it seems highly plausible that the issue exists in some form in the code.

Description: We reviewed the source code that reads the internal MBB data structures. The original report suggests that the CRC checks of the internal data blocks containing the cast vote record (CVR) and audit logs could be manipulated into reading memory regions outside the (MBB) addressable space. This would

cause invalid reads that would cause Tally to crash.

The checks for the CRC values do appear to prevent reading outside the ranges. Not only did we check these regions, but we also checked internal CVR and audit log entry CRCs, but found no case where the CRCs could be manipulated, i.e., the proper length checks were directly or indirectly made. Thus, we could not confirm that the issue indeed exists as stipulated in the original report.

We were able to confirm that the CRC function itself is unsafe; it provides only minimal checking. Thus, because it is used extensively throughout the Hart system, it is likely that such a design could be exploited to crash or otherwise subvert the Hart components.

This confirms Issue 14 reported in the California TTBR Hart source code report.

Prerequisites: Access to an MBB and control over the Tally process.

Impact: Allows the attacker to insert fraudulent votes into the voter tallies.

Procedural Mitigations: None.

Verification: This vulnerability was assessed through source code analysis and via demonstration.

19.1.2 Database passwords are stored insecurely

Boss, SERVO, Ballot Now and Tally store database passwords in configuration files on the file system. These files may be read by any user on the system.

Description: The Hart EMS applications (including Boss, SERVO, Ballot Now and Tally) store election data in databases. A username and password are required to connect to these databases. These usernames and passwords are stored in files in the same directories as the databases. An attacker able to extract these passwords can connect to the EMS databases and read and modify election, audit log and security data.

This confirms Issue 15 reported in the California TTBR Hart source code report.

Prerequisites In order to exploit this vulnerability, an attacker will need physical access to a Hart EMS system.

Impact: Possession of the username and password stored in the configuration files allows an attacker to connect to a database and read, modify and delete election data as well as EMS usernames and password hashes. Such abilities can be used to modify ballot definitions without knowing the usernames or passwords needed to login to BOSS or Ballot Now.

Procedural Mitigations: Restricting physical access to the Hart EMS systems reduces the opportunities for unauthorized users to connect to the EMS databases. Ensuring that the EMS systems are never connected to a network will also reduce such opportunities.

Verification: This issue was confirmed through source code analysis and via demonstration with the EMS databases. We located the configuration files on the EMS machines and extracted the passwords. We then used these passwords to connect to the EMS databases.

19.1.3 The EMS databases are not encrypted

The databases for BOSS, Tally, Ballot Now and SERVO are not encrypted, allowing election data to be read by any user who can open the database files associated with these applications.

Description: BOSS, SERVO, Ballot Now and Tally all store election data in databases. These databases are stored as files on the file system. Because they are not encrypted, they may be read and modified by any user on the machine.

This confirms Issue 18 reported in the California TTBR Hart source code report.

Prerequisites In order to exploit this vulnerability, an attacker will need physical access to a Hart EMS system, and to have installed a binary editor on the machine for inspecting or modifying the databases.

Impact: Because the EMS databases are not encrypted, an attacker that can open the database files can read, modify or delete election data or ballot definitions, without knowing the usernames or passwords needed to login to BOSS, Ballot Now, Tally or SERVO. This can be accomplished by opening the database in a binary editor.

Procedural Mitigations: Restricting physical access to the Hart EMS systems as much as possible reduces the opportunities for unauthorized users to open the EMS databases. Ensuring that the EMS systems are never connected to a network will also reduce such opportunities.

Verification: This issue was confirmed through source code analysis and via demonstration with the EMS databases. We connected to the databases as described in Issue 19.1.2, and read cleartext election, audit and security data.

This vulnerability was verified by connecting to the databases using the passwords extracted as described in 19.1.2, and reading cleartext election data.

19.1.4 New Users can be inserted into the database

New users with elevated privileges can be inserted into the Hart EMS databases. This allows existing users to escalate their privileges.

Description: It is important to differentiate between the usernames and passwords needed to connect to the EMS databases, and the passwords used to login to the Hart EMS applications (Ballot Now, BOSS, Tally and SERVO). A user needs the application username and password to log into the applications, while the applications use the database username and password to connect to the databases where information specific to the application (e.g., election parameters and ballot definitions in BOSS) is retrieved from. The usernames and passwords used to connect to the databases are stored in configuration files, as discussed in Issue 19.1.2. The usernames and passwords used to login to EMS applications are stored *in the EMS databases themselves*.

The Hart EMS systems use databases to store user credentials. These credentials are used to verify user identities at login. For each user, the following are stored in the database.

- Username - the name the user enters at login time
- Salt - a number used in the hashing of a password
- Password Hash - The result of a hash function performed on a password and salt

- Privileges - A number representing the privileges a user has within an application based on the username and password supplied

The method by which the password hash is generated is publicly known and well-specified,[1] and can be easily implemented in software.

Because the database passwords are not securely stored, as described in Issue 19.1.2, it is possible to connect to the databases and add usernames, salts, and hashes, allowing existing users to login to the EMS programs Ballot Now, BOSS, Tally, and SERVO with elevated privileges.

This confirms Issue 19 reported in the California TTBR Hart source code report.

Prerequisites: In order to exploit this vulnerability, an attacker will need physical access to a Hart EMS system, and to have connected to an EMS database.

Impact: An attacker able to add users to the database can create accounts using existing password hashes with escalated privileges.

Procedural Mitigations: Restricting physical access to the Hart EMS systems as much as possible, reduces the opportunities for unauthorized users to connect to the EMS databases. Ensuring that the EMS systems are never connected to a network will also reduce such opportunities.

Verification: This vulnerability was verified via demonstration. We started the database, and added new users with existing passwords, we then successfully logged in as those users.

19.1.5 Back-end Windows systems may be insecure

The Hart EMS systems, Ballot Now, BOSS, Tally and SERVO, run on the Windows operating system. An attacker exploiting a Windows vulnerability may be able to subvert these EMS systems.

Description: It is not clearly spelled out in the Hart documentation how Windows is to be configured for security on the EMS systems. The system provided by Hart had several unnecessary services turned on including remote registry editing and NetBIOS. We received machines configured with the EMS software from Hart, and verified that they ran Windows 2000 Service Pack 4 with the Update Rollup. We ran tools that showed that these machines contain a vulnerability in Microsoft Outlook Express and Windows Mail deemed "Critical" in severity by Microsoft (issue 941202).

In addition, the generation of signing keys for eCM tokens relies extensively on the `CryptGenRandom` function called by the random number generator in Windows 2000. Recent security research shows that this generator contains vulnerabilities; namely, it is possible to find previous states of the generator in about 19 seconds on a Pentium IV computer, thus allowing an attacker to determine previous signing keys. Future keys can also be determined because of the lack of both *forward security* and *backward security* in the Windows 2000 random number generator.[2]

The California source code review of the Hart system documents that Hart has procedures in place mandating that only Hart personnel may provide maintenance updates to the Windows operating system.[3] It is unclear

[1] D. Eastlake and P. Jones, *US Secure Hash Algorithm 1 (SHA1)*. Internet Engineering Task Force RFC 3174, September 2001.

[2] 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.

[3] Srinivas Inguva et al., *Source Code Review of the Hart InterCivic Voting System*. University of California, Berkeley un-

to us whether similar mandates are in place in Ohio. Regardless of the answer, there is a potential for significant difficulties. If election administrators install updates themselves, then the servers are vulnerable to inadvertent (or malicious) installation of third-party applications. In addition, it may be the case that operating system patches have unintended consequences on applications, causing them to stop functioning; an example of this was a patch for Microsoft's Internet Explorer application that caused client software from the Siebel enterprise CRM software to become unusable.[4] Election administrators may be nervous about installing patches in case a similarly unintended consequence of patch application causes Hart software to stop functioning. However, in the time needed for a patched operating system to undergo certification, the operating system, and hence the applications that run on top of it, are vulnerable to exploits.

This confirms Issue 20 reported in the California TTBR Hart source code report.

Prerequisites: In order to exploit Windows vulnerabilities, an attacker would need physical access to the EMS machines. The particular vulnerability listed requires the use of Outlook Express or Microsoft Mail, [5] which would require network access that should not be possible on these machines for procedural reasons; however, there are no technical reasons why this access could not be obtained (e.g., a USB wireless dongle could be inserted easily on the machine).

Impact: The integrity of Hart EMS applications and the election data they operate on can be compromised by compromising the underlying Windows operating system. While the current documented Windows vulnerability requires access that should not be available, the greater concern is that EMS machines need to be patched when exploits are announced, as it may be possible that some future exploit is able to cause a compromise on the machine without any additional network access.

Procedural Mitigations: Physical access to the machines running the Hart EMS applications should be restricted to only election administrators. No network interfaces should be active on the machine, and the usage of these machines must be closely supervised. In addition, either election administrators or Hart technicians must be vigilant and respond quickly to patch machines that are susceptible to newly-discovered exploits.

Verification: We found the security vulnerabilities by running the Microsoft Enterprise Scan Tool on the Hart machines. The results from this showed that the systems were susceptible to vulnerability 941202, addressed by Microsoft Security Bulletin MS07-056.

19.1.6 Election management computers may be connected to the Internet

All of the Hart back-end systems can potentially be connected to the Internet, either directly or through some other computer within a election headquarters-local network. This opens up the computer to any number of well known attacks against the operating system. The attacks could corrupt the election software or data, or introduce other malicious software that may corrupt later elections.

Description: The Ohio Revised Code 3506.23 mandates that "A voting machine shall not be connected to the Internet." This apparently does not apply to local area networks, and what constitutes a voting machine is unclear. Several counties may keep the back-end computer equipment on local area networks. However, a

der contract to the California Secretary of State, July 20, 2007 ⟨URL: `http://www.sos.ca.gov/elections/voting_systems/ttbr/Hart-source-public.pdf`⟩.

[4]Microsoft Corporation, *Microsoft Security Bulletin (MS06-013)*. October 2007 ⟨URL: `http://support.microsoft.com/kb/912812`⟩.

[5]We discovered a copy of Outlook Express runnning on the vendor-configured machine where the EMS applications were installed.

2005 Secretary of State directive[6] provides further guidance on the use of networking. This directive states that no "Election Information System" is permitted to connect to any computer network or to the Internet, including for (1) setting up elections; (2) defining ballots; (3) receiving voted data from polling places; or (4) tabulating data related to ballots or votes.

Two exceptions are (1) using local networks for creating or uploading memory cards, ballot definitions, precinct results, etc. and (2) to download software upgrades, repairs or updates (board of elections has to apply for a waiver). Under no circumstances may a polling place device be connected to an outside polling place. Modems can be used to network devices if the proper security plan is developed and permissions allowed.

While these policies are a positive step toward preventing network-based attacks, they fall short in several ways. First, modems are simply a way of networking devices. Thus, the differentiation in policy between "network" and "modem" use seems arbitrary. Any attack that could occur as a result of a network attack could also occur via modem, with the only difference being that the modem-connection is often considerably slower than wired or wireless network communications.

Second, *any* exposure of a computer to the Internet, even through other computers on the same network, is dangerous. There are many attackers on the Internet with tools that constantly scan for vulnerable hosts. It is likely that an unprotected host could be compromised within minutes of being connected to the Internet—often before it has time to download and apply the critical software "patches" that would have protected it from the vulnerability. Thus, exposing computers to the Internet even for a short period to perform maintenance is imprudent policy.

This confirms Issue 21 reported in the California TTBR Hart source code report.

Prerequisites: An attacker needs access to the Internet through which they may exploit vulnerabilities.

Impact: An attacker with network access to an EMS machine can significantly or completely determine election results by modifying the EMS applications themselves.

Procedural Mitigations: Election computers and any network to which they are connected should never be connected to the Internet. Special dispensations allowing any such connections should be eliminated. Distinctions between modems and networks in policies should be removed.

Critical system updates to the software should be certified, placed on CDs, logged, and physically installed on the computers. Note that the use of CDs *does not* prevent an attacker from corrupting the system, but only serves to prevent an attacker from remotely attacking or observing the results of compromise, i.e., this "air-gap security" is not a panacea, but prevents a wide array of common attacks. Thus, enormous care must be taken when updating election computers, and only where there is a clear and critical reason for doing so.

Verification: This issue was confirmed via discussions with Ohio election officials and through review of the relevant Ohio Revised Codes.

Note: This issue does not have any unredacted content. There is no corresponding section in the private appendices.

[6]Ohio Secretary of State. (2005). Directive 2005-23: Telecommunications: General Internet Access and Networking, Downloading Software and Modem Access.

19.1.7 eCM keys are extracted and stored insecurely

Keys used for validating the integrity of received MBBs are stored in the memory of the JBC and eScan without encryption (i.e., in cleartext). The memory of these devices can be retrieved by an attacker, and the keys can then be extracted from the memory.

Description: The eCM signing keys, used for computing HMACs for data on MBBs, are uploaded to eScan and JBC devices by SERVO, using the `NET_CMD_SET_SIGNING_KEY` command. Once uploaded, these keys are stored in the non-volatile memory of these devices in the clear. This memory can be read from a PVS device using the `MEM_READ` command, normally issued by SERVO. As described in Issue 19.4.1, the management interface used by SERVO is insecure, and thus can be used by an attacker to read the signing key from the eScan's and JBC's non-volatile memory.

This confirms Issue 24 reported in the California TTBR Hart source code report.

Prerequisites: In order to exploit this vulnerability, an attacker will need to have physical access to the eScan or JBC and have understanding of the Hart protocol.

Impact: An attacker that can read the memory can extract the signing keys used to create HMACs over data on the MBBs. This can be used to create forged MBBs which will be tallied by Tally. This attack may be performed in a matter of seconds at a polling place by anyone who can access the Ethernet interface on the eScan or the parallel interface on the JBC, including a voter.

Procedural Mitigations: Blocking or sealing the Ethernet port on the eScan and the parallel port on the JBC during an election may help to prevent an attacker from reading the device memory at the polling place. Physically securing the chassis of these devices may prevent an attacker from opening them in order to read the memory directly from the flash device.

Verification: We connected a laptop running a `memread` tool we created to extract the eScan's memory, and extracted the key from the memory image we read.

19.1.8 Vote order can be determined from an MBB

An attacker with access to an MBB can determine the order in which votes were cast. Hence, it would be simple for someone to uncover how a particular user cast his vote.

Description: The audit log and CVR (cast vote record) are two databases written to the local flash memory and MBB in the eSlate and eScan devices. The CVR contains one record for each record cast. The audit log contains entries for each important event, including the submission of a new ballot.

The CVR records are written from a large bank "slots". The first record is written to approximately the middle of the bank, and subsequent records are written to the next available slot above or below the current records based on a random flip of a coin. The audit log contains information related to the CVR. The CVR records a CRC over the unique voter ID and precinct ID. The audit log is written in sequential order and contains a time stamp.

Because the audit log contains uniquely identifying information, and because of the limited randomness of the write-from-the-middle scheme in the CVR, the order in which the votes were cast is easy to recover. This was confirmed by the reviewers in the California TTBR of the Hart system.

One additional fact pertaining to issue was uncovered during confirmation. The virtual coin in the CVR

scheme uses a deterministic random algorithm seeded by the local clock at the time the first vote is case. Because the adversary knows approximately when the first vote is cast, the sequence of coin flips can be identified.

This confirms Issue 25 reported in the California TTBR Hart source code report. The use of a poorly seeded randomness function was not previously reported.

Prerequisites: An attacker needs access to the MBB and knowledge of the pseudorandom function. This could be carried out by a poll worker or election official.

Impact: This attack allows compromise of a voter's privacy, since if the order in which they voted is known, the choices made on their ballot can be retrieved.

Procedural Mitigations: MBBs must be physically secured and access to them restricted through tight controls.

Verification: This vulnerability was confirmed through source code analysis.

19.1.9 Voting and audit data not secured while voting

The data held on the MBB (which is later used to tally election outcomes) is not secured while the polls are open. An adversary may be able to remove and alter the votes and audit logs in any way they wish.

Description: The CVR and audit log stored on the removable MBB has an associated HMAC computed over them to ensure integrity. This is referred to as a "digital signature" over the data. [7]

The data is "signed" only when the election is suspended (in the case of early voting) or when the election closes. Hence, anyone with access to an eScan or JBC could remove the card, and alter its contents. Moreover, such a modification could easily be made undetectable.

This confirms Issue 26 reported in the California TTBR Hart source code report.

Prerequisites: An attacker needs to know the format of the MBB, which could be reverse engineered with a minor effort, and have access to the MBB. Anyone who can gain access to an MBB, such as a poll worker or even a voter who is able to retrieve the card from an eScan or JBC if they are not closely supervised, can carry out this attack.

Impact: This vulnerability would allow an attacker to modify the votes occurring on an eScan or JBC.

Procedural Mitigations: Access to the MBBs must be physically controlled and access closely regulated.

Verification: This vulnerability was confirmed through source code analysis.

19.1.10 The internal vote count on the eScan, eSlate, and JBC can be modified

The Hart voting and JBC devices all maintain a counter in local memory that contains the number of votes cast on the device. The countermeasures used to protect this counter are ineffective. Thus, an attacker who is able to compromise the device can forge or reset the vote count.

Description: On the eScan, the store manager software stores the vote count value in two locations in flash memory under the identifier `STORE_VOTE_COUNTERS`. A CRC is calculated over the value and stored in

[7] This terminology is incorrect.

each of the two locations. When read, the CRC is checked and both values are compared against each other to validate that count was not corrupted. The count is only updated when the audit log is reset. The eSlate and JBC both implement essentially the same mechanisms through another vote count interface–in all other respects, such as the use of CRC and flash memory are identical.

There are two distinct counter measures intended to detect manipulated counters: redundant counters and CRCs. Neither of these solutions are in any effective against an attacker who can run executable code on these devices, e.g., by exploiting a buffer overflow. The attacker would simply replace both values in memory with those of the desired vote count, calculate the CRC of the value and place it in the CRC memory locations co-located with the vote count itself. Thereafter, there does not appear to be any way to recover the correct value.

A second attack would use the management interface to rewrite the memory regions. The `MEM_WRITE` or `MEM_WRITEFILE` commands instruct the target device to overwrite a region of memory with contents identified in the command. One could use these interfaces to simply overwrite the vote count and CRC data blocks. Moreover, the management command `NET_CMD_CLR_PRIVATE_CNT` instructs the target device to reset its internal "private" counter. Note that because of the lack of security of the management interfaces (for example, see issue 19.1.7), these commands may be easily forged by an attacker.

This confirms Issue 29 reported in the California TTBR Hart source code report.

Prerequisites: Knowledge of the memory layout of the victim device and compromise of the victim device code or access to the management interfaces is necessary. An outsider with a handheld device and access to the JBC or eScan (e.g. a voter on election day) can reset the counters within seconds.

Impact: The attacker is able to manipulate the vote counter.

Procedural Mitigations: None.

Verification: This vulnerability was verified by inspecting the source code.

19.1.11 The EMS software uses a vulnerable version of OpenSSL

Rally, Tally, and Ballot Now use OpenSSL version 0.97d, circa March 2004. Any service that uses OpenSSL is vulnerable to attacks against documented bugs in this version of the software.

Description: Rally and Tally can be set up to communicate remotely over a modem or network. Rally may be used to aggregate vote totals and communicate the vote counts to Tally, which performs the final vote tally and subsequently reports the election outcome. In addition, the Ballot Now program connects to a security database where passwords are kept over an SSL connection, which provides confidentiality by encrypting the transmissions, authentication of the other host, and integrity, ensuring that the transmission received contains the same information as what was transmitted.

All of these routines use a version of OpenSSL that is known to have vulnerabilities. The maintainers of the OpenSSL website are aware of these and have publicized them on their website.[8] We verified that an old version of OpenSSL, version 0.97d, circa March 2004, is in use in the Hart system, that and Rally, Tally, and Ballot Now use routines from this library.

This confirms Issue 31 reported in the California TTBR Hart source code report. The reliance of Ballot Now on OpenSSL and the vulnerable version it uses was not previously reported.

[8] ⟨URL: `http://www.openssl.org/news/vulnerabilities.html`⟩.

Prerequisites: An adversary who has knowledge of the attacks possible against OpenSSL 0.97d, including the ability to craft a buffer overflow attack or causing the machine connecting to a server to crash, could make use of these vulnerabilities. Since this is publicly available knowledge, anyone with access to the affected Hart applications (Rally, Tally, Ballot Now) may implement this attack.

Impact: An attack can cause servers to crash, hampering collection of election results or possibly preventing users from logging into applications necessary for running the election.

Procedural Mitigations: Rally and Tally are not allowed to connect remotely in Ohio, but are allowed to operate over a "local" network. Access to this network must be very closely guarded and monitored. Optimally, the machines running these services should have their network and modem interfaces removed.

Verification: The vulnerabilities in OpenSSL v0.97d are publicly known and listed on the website of the software maintainers. The vulnerability was verified through multiple methods of determining the library version.

19.1.12 Pervasive failure to follow safe programming practices

The Hart system implementation does not follow commonly accepted practices for safe source code development. This has caused, and will continue to lead to, frequently occurring security and reliability issues such as those reported throughout this and prior Hart analyses.

Description: There are a multitude of exploitable errors caused by poor coding practices; buffer overflows, `printf` attacks, integer overruns, unchecked or inconsistently checked and propagated error conditions, poor compartmentalization of data and functionality, inconsistent and over-modularization of libraries, improper trust, poor memory and resource management, and dangerous uses of operating system services (e.g., the Windows registry) are persistent throughout the code. Such errors are the bread and butter "hooks" that an attacker uses to subvert a system. We consider coding practices and make recommendations for future activities in 18.4, and defer further discussion to that section.

The Hart Source Code Report in the California TTBR study outlines the ways that the code failed to meet coding practices. We point the reader to that report for deeper exposition on the issues raised by these practices. In all cases, we found the failures pointed out in the Hart were present. Note that issue 36 of the TTBR report is encompassed by the larger design, development, and maintenance issues identified in Chapter 18.

This confirms Issue 36 reported in the California TTBR Hart source code report.

Prerequisites: The means of exploiting the above vulnerabilities are well known and there exist many tools to assist the attacker.

Impact: These errors lead to an exploitable and unstable implementation.

Procedural Mitigations: None

Verification: This vulnerability was confirmed through source code analysis and observationally through the study of this and numerous other issues.

19.2 Hart eCM Vulnerabilities

19.2.1 The same symmetric eCM key is used county-wide

Only one symmetric key is used by the eCM throughout the entire jurisdiction. This key is protected by a PIN that may be the same for each eCM token, and is propagated to every device. If an attacker can compromise this key, they can perform actions such as forging MBBs.

Description: All message integrity in the Hart system is provided by a message authentication code (MAC) key. This is the same throughout the entire county, though Hart indicates that multiple keys may be used. Before an election is called, a single PIN is generated by the appropriate election official, and this is used as the basis for generating the eCM tokens in the eCM Manager program. Only one key may be associated with a key. In eCM Manager, all eCM tokens used in the county are written, commonly using the same PIN, [9] and have the same symmetric key and key ID installed on them. The key on an eCM is then used by BOSS to generate MBBs, and by SERVO, which propagates the key to an eScan or JBC over a wired connection. In this fashion, all devices will be able to validate data that has an HMAC associated with it.

This confirms Issue 22 reported in the California TTBR Hart source code report.

Prerequisites: A poll worker who has access to the PIN can extract the key from the eCM, retrieve the information from an obfuscated file written out by eCM Manager (see Issue 19.2.2), or attempt to extract the key from a JBC or eScan as described in Issue 19.1.7. If any of these attacks are successful, the poll worker will have the key for the entire county.

Impact: An attacker may be able to forge MBBs and cause an arbitrary ballot to appear. This could affect the way a voter casts their vote by having them choose an unintended candidate, and may affect the final vote totals when the MBB is read into Tally.

Procedural Mitigations: It may be possible for different PINs to be associated with different keys and to partition the hardware in such a way that only if a particular PIN is entered will an intended key be used. Thus, MBBs could be written that will only work in eScans or JBCs respectively, or different PINs could be used for different precincts or regions within a county. This would require using the correct PIN each time the eCM is used for each different precinct/region.

Verification: These vulnerabilities were verified by source code inspection.

Note: This issue does not have any unredacted content. There is no corresponding section in the private appendices.

19.2.2 eCM keys are stored insecurely on the eCM manager

The eCM Manager software allows users to store the contents of an eCM cryptographic token to file. The file is "obfuscated" in an insecure fashion on the machine or device where the information is being stored. The obfuscation consists of performing the XOR operation on each character with the letter 'x', a mechanism that can be easily defeated.

[9] Hart suggests in the *Hart Voting System Management & Tasks Training Manual 6300-001 62E* that using multiple PINs will increase security at the costs of needing to specially mark and track PINs for each copy of the eCM. Anecdotal evidence suggests that a single PIN is commonly used.

Description: eCM Manager files are stored by default with the file extension '.eCM'. These files can be easily searched for on a machine. A simple program can be written to perform the XOR operation on each character with the letter 'x' that will recover the original details of what is stored on the eCM token.

This confirms Issue 23 reported in the California TTBR Hart source code report.

Prerequisites: The attacker needs to have access to the machine or device that the .eCM file is stored on. The attacker must know that an XOR encoding has occurred, although this is not difficult to determine for a skilled attacker. A simple program can be written to recover the original data in the file that corresponds to information on the eCM token.

Impact: This vulnerability allows an attacker to know the value of the secret key stored on the eCM token, which allows for further attacks such as forging incorrect ballot details on an MBB.

Procedural Mitigations: Access to the machines running the eCM Manager must be closely monitored. There are few cases where the file should be allowed to be saved and these should be closely monitored.

Verification: This issue was confirmed through via demonstration with the '.eCM' files produced by eCM manager. We reversed the obfuscation technique to extract the eCM keys.

19.3 Hart SERVO Vulnerabilities

19.3.1 Buffer overflows in SERVO

SERVO contains buffer overflows that can allow an attacker to execute arbitrary code.

Description: SERVO receives data from the eScan, eSlate and JBC via the "Net Message" structure. This structure contains a block of data, and a header containing the length of the block. SERVO copies the full data block into its own fixed size buffer without checking the length supplied in the header, allowing a block of data larger than the allocated buffer to be copied. This can result in arbitrary code being injected into SERVO and executed.

These buffer overflows appear in code for reading, writing and verifying the firmware and audit logs on eScan, JBC, and eSlate devices. Some of these buffer overflows exist in functions not reachable by any execution path, but are indicators of insecure programming practices.

This confirms Issue 13 reported in the California TTBR Hart source code report.

Prerequisites: In order to exploit this vulnerability an attacker will need to have compromised an eScan, eSlate, or JBC device. We show in Issue 20.3.1 that the firmware may be completely replaced in under two minutes on an eScan.

Impact: A buffer overflow allows an attacker to execute arbitrary code, and thus, completely control SERVO. This could lead to compromised firmware images or configuration files being uploaded to PVS devices while SERVO is used to perform administrative operations on those devices.

Procedural Mitigations: None

Verification: This vulnerability was verified by inspecting the source code.

19.4 Hart eScan Vulnerabilities

19.4.1 The eScan is managed via an accessible Ethernet port

The eScan optical scan device can be remotely administered via an Ethernet port, located at the back of the unit. An attacker able to access this port can perform management operations on the eScan such as replacing its configuration file or reading its memory.

Description: eScan listens on port 4600 for TCP connections. eScan normally uses a default IP address which is located in its configuration file on the eScan. Normally, this port is used for sending and receiving messages to SERVO. The protocol shared between eScan and SERVO includes commands to perform the following functions:

- Sending and receiving files from the eScan
- Reading the image of the eScan's non-volatile memory
- Uploading the eCM signing key
- Uploading and installing a new configuration file or eScan executable

Neither machine authenticates to the other, making it easy to impersonate SERVO to the eScan, or vice versa.

Note that the communications protocol between SERVO and the eScan unit is the same as that used between SERVO and the JBC, and between the JBC and eSlate units. Thus, knowledge of the protocol between any of these devices yields protocol details for all of them. Reverse-engineering the protocol between the eScan and JBC is facilitated by the fact that data is communicated over an Ethernet interface. Placing a laptop between the two devices to "sniff" the interface would allow that machine to garner details of the communications, and many tools are available to decode protocol details over Ethernet, such as the open source network protocol analyzer *Wireshark*.[10]

This confirms Issue 3 reported in the California TTBR Hart source code report.

Prerequisites: In order to leverage this vulnerability, an attacker will need physical access to the eScan and have an understanding of the Hart protocol. An attacker with access to the Ethernet interface and a handheld device running a tool which can issue commands over this interface can compromise an eScan in seconds.

Impact: Because the protocol allows for the sending and receiving of arbitrary files from the eScan, an attacker that can leverage this vulnerability may completely control the operation of the eScan. It is also possible to read the eScan's memory and extract the eCM signing key as described in Issue 19.1.7.

Procedural Mitigations: Blocking the Ethernet port on the eScan during an election may help to prevent an attacker from being able to command it in the polling place.

Verification: This issue was confirmed through source code analysis and via demonstration with the eScan. We created a set of tools allowing us to manipulate the eScan via its Ethernet port. Namely, the `getfile` and `upload` tools allowed us to retrieve, upload and install the eScan firmware and configuration file, and our `readmem` utility was used to read the eScan's non-volatile memory.

[10] *Wireshark*. ⟨URL: `http://www.wireshark.org`⟩.

19.5 Hart eSlate Vulnerabilities

19.5.1 The eSlate is managed via a serial port connection to the JBC

The JBC manages the eSlate over a serial connection whose interface is located at the top of the eSlate. The eSlate does not attempt to verify that the commands it receives over this interface are from the JBC. Thus, an attacker may connect a device to the eSlate over this serial connection and control it as the JBC would.

Description: The JBC sends control messages to the eSlate over the serial port on the eSlate. The eSlate runs a "Net Task" that waits for messages from the serial port and processes them. At no point does this task try to verify that the messages it receives are from the JBC. The commands this task will execute include the following:

- Writing to the internal audit log
- Clearing the internal CVR and audit logs
- Getting CVR records from the eSlate
- Getting the status of generated voter codes
- Reading and writing the eSlate's memory

An attacker that can connect to the eSlate's serial port can impersonate the JBC and perform any of the above listed commands.

This confirms Issue 2 reported in the California TTBR Hart source code report.

Prerequisites: In order to exploit this vulnerability, an attacker will need physical access to the eSlate, and have an understanding of the Hart protocol. An attacker using a handheld device and a tool that emulates the JBC, can impersonate the JBC to the eSlate.

Impact: An attacker able to communicate with the eSlate may control the eSlate to the same extent that the JBC may. This includes the ability to read, modify and delete the CVR and Audit logs, and change the eSlate mode, e.g. polls open, polls closed etc. These abilities can be used to modify or destroy the election results from an eSlate and infringe on voter privacy.

Procedural Mitigations: Placing a tamper proof seal between the eSlate and the attached serial cable could make it evident if the cable were detached.

Verification: This issue was confirmed through source code analysis.

19.5.2 Communication between the eSlate and JBC is insecure

The JBC and eSlate communicate via the serial port interface mentioned in issue 19.5.1. The uses of this channel include eSlate management, the transfer of ballots from the JBC to the eSlate and of CVRs from the eSlate to the JBC, and the validation of voter access codes. Because no authentication or encryption of data is used, an attacker may interpose himself between the JBC and eSlate and either eavesdrop on traffic between the JBC and eSlates or impersonate any of these devices.

Description: No cryptographic measures are taken to ensure either authentication, message integrity or message confidentiality. This means that an attacker can impersonate either the JBC or the eSlate, or eavesdrop on the communication over this channel. Impersonating the JBC allows an attacker to do the following:

- Acknowledge the reception of CVRs, thus preventing the JBC from recording them
- Control the eSlate, as described in Issue 19.5.1
- Claim that invalid voter codes are valid, thus preventing a CVR from being recorded

Impersonating an eSlate allows an attacker to do the following.

- Send forged CVRs to the JBC
- Mislead the JBC into believing that the management operations described in Issue 19.5.1 were carried out, when they were actually ignored

This confirms Issue 5 reported in the California TTBR Hart source code report.

Prerequisites: In order to exploit this vulnerability, an attacker will need to have physical access to the serial cable connected to the top of the eSlate or at the back of the JBC, and have an understanding of the Hart protocol.

Impact: An attacker able to eavesdrop on the JBC to eSlate connection or impersonate an eSlate or JBC may, read, modify or destroy election records, prevent votes from being counted, and infringe on voter privacy by monitoring votes as they are cast.

Procedural Mitigations: A tamper evident seal could be placed between the serial port connector and the eSlate, as well as the serial port connector and the JBC. Note that this would only make it evident that an attack had occurred, and would not reveal any information about the nature of the attack or to what extent it impacted the election results.

Verification: This issue was confirmed through source code analysis.

19.5.3 Compromised eSlates can provide votes without voter records

The JBC does not verify that the eSlate provides only one vote per voter code. Essentially, the eSlate is charged with performing checks to see that the voter code is correctly processed. Hence, a compromised eSlate can ignore such checks and simply "stuff" the ballot box with votes.

Description: The JBC polls the eSlates for status at a regular interval. Included with the response is a status flag indicating, among other things, that a) the local voter code needs to be validated, and b) whether there is a CVR record ready to be processed. A compromised eSlate can simply signal (b) every poll and thereafter insert votes into the record.

Moreover, the compromised eSlate can listen in on the communications between the JBC and other eSlates to extract voter codes, or predict voter codes before they are issued. If properly timed, the additional inserted votes will be difficult to identify after the fact.

This confirms Issue 8 reported in the California TTBR Hart source code report.

Prerequisites: An attacker needs to have compromised one of the eSlates used in the election.

Impact: A compromised eSlate will allow the attacker to hijack voter codes and use them to vote in an arbitrary manner.

Procedural Mitigations: Prevent the eSlate program from being compromised.

Verification: This vulnerability was confirmed through source code analysis.

19.5.4 The periodic memory integrity checks used by eSlate and JBC are ineffective

Both eSlate and JBC devices check the integrity of blocks of internal memory used every 10 seconds. However, these checks are trivially bypassable. Thus, these checks are very unlikely to detect a system compromise.

Description: A mechanism often used to check data integrity is a *cyclic redundancy check* (CRC). This is commonly used to detect errors in communication or within blocks of data on persistent storage or memory devices. A CRC function works by calculating a mathematical function over the digital values stored in each byte of the checked region. The resulting value (2 bytes in the case of the Hart implementation) is called the *CRC value*. Later, the integrity of the target device is checked by recomputing the CRC value over the same data. If one or more bits of data have changed, it is highly likely that the CRC value will change.

Both the eSlate and JBC execute a background thread that performs a number of service functions. One of these functions detects when memory regions have been changed. Each of the devices maintains a table of "important"[11] data regions and associated CRC values. Every 10 seconds, the processor recalculates the CRC of the tested regions and compares it against the values stored in the table. If the CRC values do not match, a error is generated and the system fails.

One problem with CRCs is that they are not secure. An adversary who has control over the value in the region can easily craft a new (and different) set of data values that calculate a new set of data values that compute to the same CRC.

This approach is unlikely to be effective in detecting or preventing malicious activity. Here are but a few of the reasons for this conclusion:

- An attacker can penetrate the system and replace the existing data/code with its own such that it calculates to the same CRC.
- An attacker can replace the code that executes the check with some others that perform some other function, or none at all. Simply replacing the call with `noop` (null operation) codes would be a quick and easy way to patch the existing code.
- An attacker can replace the data held in the table to some known region. For example, there is nothing in the source code to prevent the table from simply checking itself over and over.
- An attacker could patch the firmware, but not the table in memory. This would cause a crash, but the device would run the new firmware when it boots.
- An adversary could use this list in conjunction with buffer overflows to change data values in one of the checked areas. If carefully placed, the failure would only be detected at the next CRC check sequence. Because of the potentially long period between the malicious input and check, the adversary could mask his behavior.

[11] It is unclear what policy is used to determine what regions of memory are deemed important.

Note that 10 seconds is very long time in computer terms. Literally billions of instructions are executed even on modest hardware in a single 10 second period. Thus, the adversary could launch a very large number of activities in this period.

This confirms Issue 9 reported in the California TTBR Hart source code report.

Prerequisites: In order to exploit this issue, an attacker will need knowledge of the memory layout of the victim device, and access to the device to upload firmware.

Impact: If an attacker has compromised an eSlate or JBC device, it will likely be undetected.

Procedural Mitigations: None

Verification: This vulnerability was verified by inspecting the source code.

19.6 Hart Servo Vulnerabilities

19.6.1 SERVO-based device firmware checking can be spoofed

SERVO can verify the integrity of the firmware on a JBC, eSlate or eScan by comparing the SHA-1 hash of that device's firmware image to a Hart supplied hash. Non-matching hashes result in an error. SERVO depends on the device in question to provide its firmware image, and thus, a compromised device may simply store an uncompromised copy of the firmware image, and provide it to SERVO as prompted.

Description: For each device to be inspected, SERVO first obtains that device's ID and checks to see if it is already in a database of devices for a given county. If it is not, SERVO will obtain the device's firmware version number, consisting of major, minor and release fields, and compare it against the version number stored in its database. Next, SERVO requests the device's firmware image, which it computes a SHA-1 hash over, and compares against the hash in its database.

This method for device firmware verification assumes that the device will provide the correct firmware image. This is not necessarily the case. A compromised device could store an uncompromised copy of the firmware, and provide it to SERVO when prompted.

This confirms Issue 11 reported in the California TTBR Hart source code report.

Prerequisites: In order to exploit this vulnerability, an attacker will need to have compromised a JBC, eSlate or eScan.

Impact: If an attacker has compromised a JBC, eSlate or eScan, this vulnerability allows the compromise to go unnoticed by SERVO. This creates the possibility that a compromised JBC, eSlate or eScan could be re-used in future elections without the compromised firmware being noticed.

Procedural Mitigations: None.

Verification: This issue was confirmed through source code analysis.

19.7 Hart JBC Vulnerabilities

19.7.1 The JBC internal version checks can never fail.

The JBC checks its own version number on startup. This check can never fail, and thus provides no meaningful security function.

Description: A statically defined version number is compiled into the code, but it is never checked. A function apparently exists to compare this static version against some legitimate numbers, but this is not implemented. Thus, the functionality of the call is suspect. While this is not a vulnerability on its own, it is indicative of incomplete functionality that is pervasive throughout the system, as discussed further in chapter 18.

This confirms Issue 10 reported in the California TTBR Hart source code report.

Prerequisites: None

Impact: None

Procedural Mitigations: None.

Verification: This vulnerability was confirmed through source code analysis.

19.7.2 JBC-based eSlate firmware checking can be spoofed

The JBC attempts to verify the integrity of attached eSlate devices by performing a CRC check against the eSlate and checking its version number. This approach requires the eSlate to be answering truthfully, which may not be the case if the device has been maliciously altered.

Description: When the JBC is started, it finds eSlates attached to it and retrieves information associated with them. It in turn assigns a node ID value to address the machines. A value identified as a CRC value in the code is retrieved, as is the version number of the eSlate. As a test of the eSlate's integrity, the major version number of the device is compared to a hard-coded value that the JBC possesses; the other parts of the version number (minor number) and CRC are not checked at this time. If the verification of the major number succeeds, the eSlate passes the test. This value is given by the eSlate, meaning that if a malicious device has provided incorrect numbers to the JBC to disguise itself as a legitimate eSlate, the JBC will not detect this. In addition, the check only occurs at startup, so an eSlate that is compromised during an election will not be rechecked.

This confirms Issue 12 reported in the California TTBR Hart source code report.

Prerequisites: An attacker will need to have compromised an eSlate device to leverage this vulnerability. This could potentially be performed by a poll worker.

Impact: If an attacker has compromised an eSlate device, this vulnerability allows the device to mask itself by passing the integrity check, and then act without concern of being detected.

Procedural Mitigations: Strong physical security and controlled access to the eSlate is necessary to attempt to prevent attacks that may compromise the machine.

Verification: This vulnerability was verified by inspecting the source code.

19.7.3 The JBC is managed via an accessible parallel port

The JBC is managed by the SERVO application over a parallel port in the back of the unit labeled "printer". An attacker that can access this port can control the JBC and any connected eSlates.

Description: SERVO connects to the JBC over a parallel port to perform administrative tasks. These tasks include the following.

- Uploading the signing key
- Controlling the JBC's LCD display
- Clearing the CVR logs and audit logs

No authentication of the device acting as SERVO is performed by the JBC. Thus, an attacker with any device with a parallel port interface can issue any and all commands to the JBC and eSlates that SERVO can. We describe our utility and a separate attack that may be used to fully delete the results of an election in Section 20.4.2.

This confirms Issue 1 reported in the California TTBR Hart source code report, which identified the vulnerability in source code, but did not perform any verification.

Prerequisites: In order to exploit this vulnerability, an attacker will need to have physical access to the JBC and an understanding of the Hart communication protocol.

Impact: An attacker controlling the JBC over this interface can perform the same administrative functions as SERVO, including those listed above.

Procedural Mitigations: Securely blocking access to the JBC's parallel port, such as with a lock or a seal, at all times except when it is being administered by SERVO, will be a step towards preventing unauthorized devices from being connected to the JBC, though the utility of this mitigation may be limited in the face of a determined attacker with the correct tools to circumvent the physical countermeasures.

Verification: This issue was confirmed by source code analysis and via demonstration with the JBC. We wrote a `jbccmd` program which we ran on a laptop connected to the JBC's parallel port. Using this program we successfully issued commands to the JBC and eSlate with no authentication.

19.7.4 The JBC Voter Registration Interface can be used to generate voter access codes

A modem interface, accessible with a DB-9 cable, is located on the back of the JBC. This interface is serial and is known as a VRI, or Voter Registration Interface. A VRI device attached to the JBC can be used to send instructions to generate access codes for voters. An attacker can use this port to generate their own codes to be used for voting on the eSlate machines.

Description: In early voting mode, the VRI may be used to generate access codes. These may be printed from the JBC's printer or printing may be suppressed. In either event, as long as a properly formatted request is issued over the VRI to the JBC, it will return a valid access code. Furthermore, when the JBC is in general election mode and the polls are marked as open, the VRI assumes the existence of a barcode scanner that can be interfaced with the JBC. Commands can be issued through this interface that allow access codes to be printed out on the JBC's printer; an arbitrary number of these codes may be generated, and they are valid for a length of time determined when the election is defined in BOSS. While the copy of BOSS sent to us

defaults to 30 minutes as a timeout period for the access codes, this value can be set as high as 960 minutes (16 hours). [12]

In a further result to what was noted about this vulnerability in the California TTBR Source Code report, we found that despite the documented limit of 150 outstanding voter access code requests, we were able to generate many more valid codes; we were, in fact, able to generate over 10,000 access codes within a single expiration period, meaning that duplication in codes must have occurred. At this point, any code entered into the eSlate, as long as it had not already been used, would be valid for voting with. One note is that the number of active codes is displayed on the JBC, and as the number passes 500, the value displayed turns to '____'.

This confirms Issue 4 reported in the California TTBR Hart source code report.

Prerequisites: The attacker must be able to access the serial port on the JBC and possess a device, such as a PDA, to run a terminal emulator program. They must also know the precinct ID.

Impact: A voter would be able to vote an arbitrary number of times before the access codes expire.

Procedural Mitigations:

We were not supplied with any devices to leverage the VRI, which leads us to suspect that devices such as voter registration equipment or barcode scanners are not meant to be plugged into the JBC for elections in Ohio. Accordingly, the serial interface on the JBC should be sealed off or removed altogether, as it has no purpose. There are two models of JBC that are manufactured by Hart, the JBC 1000 B, which has a modem, and the JBC 1000, which does not.[13] We received a JBC 1000 B for testing purposes with a serial port blocker from the Hamilton County Board of Elections. This indicates that the JBC 1000 B is used in the field in Ohio. The port blocker is attached to the modem port on the JBC with two screws that may be easily removed by an attacker, and the security it provides is minimal, if not cosmetic. Only JBC 1000 models, which do not contain a modem, should be used in Ohio.

Verification: This vulnerability was confirmed through source code analysis and demonstration with the JBC. We wrote a program to retrieve voter codes over the VRI in early voting mode, and to issue code requests on election day through the same interface as our program pretended to act as a barcode scanner. We verified that over 10,000 codes could be active at a given time by printing an access code report from the JBC.

19.7.5 Format String vulnerabilities in the JBC report mode

The implementation of several reports available to poll workers at the close of the polls has errors that can be exploited to extract arbitrary data from the JBC. This attack may be used to extract key data from the JBC or to execute other code.

Description: Poll workers have the option of generating three kinds of reports after closing the poll: a voter code summary report, a vote tally report, and a write-in report. The implementation does not perform *input filtering* on the write-in data, meaning it is possible to enter code disguised as regular input text. Specifically, code on the JBC allows user-supplied data to be interpreted as `printf` format strings. These allow the user to extract data from the stack or heap and print them on the reports, and possibly to execute other code.

[12] Hart Intercivic, *BOSS Operations Manual 6100-019 Rev. 43-62A*, page 489.

[13] Hart Intercivic, *Hart Voting System, System for Election Records and Verification of Operations: Operations Manual, Software V. 4.2, 6100-102 Rev. 42-62B (SERVO Operations Manual)*..

This confirms Issue 6 reported in the California TTBR Hart source code report.

Prerequisites: Access to a vote operation on the JBC and a method of inputting special characters necessary to mimic `printf` command statements. While these are not generally accessible through the eSlate's on-screen keyboard, it would be possible to send these commands if the eSlate was compromised (Issue 19.5.1) or if the broadcast network between a JBC and eSlate was tapped such that commands could be sent over it (Issue 19.5.2).

Impact: All secret data may be extracted from the JBC, and arbitrary code could potentially be run on it. This could be an attack vector for loading malicious code onto the JBC, for example, as a means of infecting SERVO. [14]

Procedural Mitigations: Restrict access to the JBC and all eSlates to minimize the chance of a compromised host.

Verification: This vulnerability was confirmed through source code analysis.

19.7.6 Voter codes are not selected randomly

All past and future voter codes for a precinct can be determined from a single voter code. Anyone with access to a single voter code (a legitimate voter, for example) can easily determine the voter code of everyone who has voted or will vote at that precinct.

Description: Four-digit numerical voter codes are used to authenticate voters using an eSlate at a polling place. After receiving the number from a poll worker, the voter enters it at an eSlate to begin voting. The eSlate checks with the JBC to ensure that the voter code was issued and that it has not been used before.

Voter codes are selected by selecting a random number from 0-9999 and performing a permutation on the "index" value. Subsequent voter codes are determined by adding one (wrapping from 0-9999) to the index and permuting as before. Because the permutation is invertible and constant, anyone with access to one voter code can compute the sequence.

This confirms Issue 7 reported in the California TTBR Hart source code report.

Prerequisites: Knowledge of the algorithm used, which has been reverse-engineered without source code.[15] It should be assumed that this algorithm is known, as it can be easily recovered from anyone who has access to a working JBC or can see a sequence of values, e.g., any poll worker handing out voter codes.

Impact: This vulnerability allows an attacker to cast multiple votes, potentially without detection, and to prevent others from using that same vote.

Procedural Mitigations: Only one eSlate should be used in a polling place, and only one voter code should be allocated at any one time. This is the only effective way to prevent multiple voting and stealing access codes from other voters.

Verification: This vulnerability was confirmed through source code analysis. In addition, we developed a utility that will print the next series of access codes that will be generated given a code, as well as previous codes in the sequence. We tested this by receiving an access code from the JBC and running the utility with the given code as input. We verified that the sequence of access codes generated were the same as those

[14] See Section 21.1 for more information on this attack.

[15] Elliot Proebstel et al., 'An Analysis of the Hart Intercivic DAU eSlate'. In Proceedings of the USENIX/ACCURATE Electronic Voting Technology Workshop. August 2007.

issued by the JBC.

19.8 Hart VBO Vulnerabilities

19.8.1 The VBO record is easily manipulatable by an eSlate program

The VBO record is printed using a rudimentary printer. This printer is completely under control of the program to which it is attached. This allows a compromised eSlate to prevent any use of the VVPAT, to invalidate legitimate votes, and to insert illegitimate votes.

Description: The printer used by the eSlate is under control of the eSlate software. The printer software simply follows commands provided by the eSlate and executes them. Two functions, `Print` and `FormFeed` allow the eSlate to arbitrarily modify the VVPAT record, often without the knowledge of the voters.

An attacker can use this function to invalidate legitimate ballots by printing a `BALLOT REJECTED`, then printing `BALLOT ACCEPTED` with votes of the attackers choice.

Finally, the printer can be forced to forward the paper to the end of the reel, thus disabling printing.

This confirms Issue 33 reported in the California TTBR Hart source code report.

Prerequisites: An attacker must compromise the eSlate and upload new firmware either to the eSlate or VBO.

Procedural Mitigations: Because limiting physical access to the eSlate and VBO is not possible, the usage of tamper evident seals would at least reveal if the VBO or eSlate's serial port interface has been tampered with.

Verification: This vulnerability was confirmed through source code analysis.

19.8.2 VBO printer allows the paper to be rewound

The VBO record is printed using a rudimentary printer. The interface allows the controlling software to instruct the printer to rewind the paper. It may be possible for an adversary to use this feature in the VBO software to overwrite legitimate ballots.

Description: The printer has an I/O pin that, when raised, appears to instruct the printer to rewind the paper. If the printer software is compromised, it could trigger the reverse motor after the ballot is completed. This may cast doubt on the ballot and possibly make its content unreadable.

Note that this code is compiled via MPLAB embedded processor tools. One would need to exploit for example, a buffer overflow, to take control of the processor and execute commands to perform this attack. There do not appear to be any commands that allow the user to exploit this feature without taking over the firmware.

This confirms Issue 34 reported in the California TTBR Hart source code report.

Prerequisites: Access to the printer driver software.

Impact: Allows the attacker to invalidate or corrupt the VVPAT record generated by an eSlate system.

Procedural Mitigations: Prevent application printer driver program from being compromised.

Verification: This vulnerability was confirmed through source code analysis.

19.8.3 VBO ballots are printed sequentially

The VBO record is printed in order on a single roll of printer paper. An attacker with access to the voter record book can therefore correlate the vote on the roll to the individual voter.

Description: The eSlate only allows printing of a ballot to occur on the roll of paper. Thus, the order of voting is recorded implicitly from the order in which it is recorded. Anyone who has access to the voter signin or can observe the order in which known voters voted can determine the way in which they voted.

This confirms Issue 35 reported in the California TTBR Hart source code report.

Prerequisites: Access to the printer voter rolls or observation and VVPAT.

Impact: An attacker able to correlate a vote on the VVPAT to an individual voter can compromise that voter's privacy.

Procedural Mitigations: Restrict access to VVPATs and voter records.

Verification: This vulnerability was confirmed through source code analysis and by observation.

19.9 Hart Tally Vulnerabilities

19.9.1 The Tally interface allows a Tally administrator to "adjust vote totals"

Tally allows an election administrator to manually adjust the vote totals for a jurisdiction, to allow for voting sites where no Hart equipment is being used. While an audit log record is created, there are no other records made that indicate an adjustment has been made, and there is no history of adjustments displayed in case an error in entering the totals is made.

Description: Adjusting the vote total will add to the total number of votes captured in the database; there is no distinguishing these votes from those that were counted through the Hart equipment. Rather than showing the adjustments to the vote total, the total is overwritten outright in the database. This can cause errors during a recount or, if a recount is not called, the changes may go unnoticed, allowing for the election results to be tampered with.

This confirms Issue 17 reported in the California TTBR Hart source code report.

Prerequisites: Accessing the "adjust vote" feature is only possible by an administrator on the system. If an administrator is malicious, or if someone has access to the system and can guess the administrator username and password, they can alter the vote counts. Note that any user with physical access to the EMS machines can login to the EMS applications with administrator privileges as described in Issue 20.1.3.

Impact: The number of votes can be altered for an election. If a number has been incorrectly entered, it is unclear how many votes should be reversed as no history of adjustments has been shown; this can be a source of confusion to election officials who may end up making incorrect adjustments. In addition, while the adjustments are logged in the audit log, other vulnerabilities have shown that this log can be modified

(particularly Issue 19.1.3) by modifying the databases; thus, it is possible for a malicious user to both adjust totals and cover their tracks.

Procedural Mitigations: A potential procedure to mitigate this threat is for a to observe an administrator adjusting the vote totals, and logging this value on paper. In addition, access to the machine running Tally must be secured against any user without a corresponding witness observing all activity.

Verification: This vulnerability was confirmed through source code analysis. We also ran Tally and verified that when the manual adjustment is made, there is no on-screen indication of the number of adjusted votes.

19.9.2 Precinct IDs are improperly handled by the system

MBBs can be manipulated to force the Tally software to accept votes for precincts that that MBB is not authorized for. Manipulation of the Tally database can prevent votes from being being counted.

Description: Every MBB is authorized to report votes for a precinct identified in its internal data. This information is recorded in an election database created by BOSS and subsequently used by Tally for reconciling the MBBs. Cast vote record (CVR) data tagged with precinct identifiers not associated with the card in the database are ignored. MBBs encoded as being SERVO recount MBBs (designated by an equipment identifier in the MBB header) are an exception to this. These cards are implicitly authorized to report data for all precincts.

Because the equipment identifier is read from the header, an attacker can create a card listing votes from any number of legitimate precincts simply by changing the identifier to SERVO. Tally will see the SERVO identifier and unquestioningly add the votes to the election results.

Further, an adversary who can manipulate the Tally database can change the precincts associated with a legitimate MBB. When the legitimate MBB is processed by Tally, the votes counted are ignored.

This confirms Issue 27 reported in the California TTBR Hart source code report.

Prerequisites: An attacker must know the format of the MBB, which could be reverse engineered with a minor effort, and have knowledge of the MBB key.

Impact: This vulnerability may allow an adversary to add votes to an arbitrary precinct, or to prevent legitimate votes from being counted.

Procedural Mitigations: MBBs must be well-protected.

Verification: This vulnerability was confirmed through source code analysis.

19.9.3 Users can tally unclosed or corrupted MBBs

Failure of the integrity check software can be ignored by the user when processing MBBs. In short, the user can ignore error messages and introduce potentially corrupt, modified, or forged voting data into the Tally software.

Description: The Tally software checks the cryptographic integrity when an MBB is inserted into the local drive by calculating an HMAC. If this check fails, the user is presented with a message “The MBB data fails cryptographic validation. Accept the MBB?” and buttons for “yes” and “no”. If the user clicks the “yes” button, then the MBB is processed as if the integrity check did not fail.

This feature could be used by an attacker to insert fraudulent votes into the database. The attacker who gains access to the MBB during or after an election, but before it is tallied, can create cast vote record (CVR) entries that appear to be legitimate. A malicious election official, or simply one who ignores the warning and accepts the MBB, will introduce invalid votes.

Tally creates two events when the HMAC failure ("MBB digital signature failure") is detected and MBB is subsequently accepted ("MBB User Accepted"). This is the only indication that the corrupt MBB was used.

This confirms Issue 28 reported in the California TTBR Hart source code report.

Prerequisites: Access to an MBB and control over the Tally process. The attacker would also need a program capable of reading the MBB and writing the CVR record onto it.

Impact: This attack allows the attacker to insert fraudulent votes into the voter tallies.

Procedural Mitigations: Access to the MBBs must be closely controlled. The tallying process must be carefully monitored, and audit logs must be carefully reviewed after the election is over.

Verification: This vulnerability was confirmed through source code analysis and via demonstration. The review team created a program to duplicate previously cast votes in the MBB CVR log. The HMAC was not changed, and an error was correctly detected. The fraudulent votes were accepted and reflected in the vote tallies.

19.9.4 MBBs are only processed if the Tally database allows it

Tally records in its internal database the unique identifier of all MBBs that it has processed. Tally will not record votes from an MBB that is already in this list of "tallied" cards. An attacker who can either modify the internal Tally database or provide forged MBBs to Tally can prevent legitimate MBBs (and thus votes) from being tallied.

Description: Tally keeps a table of processed MBBs in its internal database, which is keyed by the unique ID of each MBB. A MBB's votes are tallied only if a record is not in the processed MBBs table. Specifically, when an MBB is inserted and tallying is attempted, Tally checks to see if a record in the above table associated with the unique ID exists. If no record exists, the MBB votes are added to the election tallies. If a record does exist, the MBB will be rejected.

Legitimate MBBs can be prevented from being tallied in at least two ways. First, an attacker with access to the database can simply insert records containing the IDs of MBBs that they do not wish to be counted. Second, an attacker with access to the county-wide key (see Issue 19.2.1) can create MBBs with the same IDs as the legitimate ones it would like to prevent from being processed.

This confirms Issue 30 reported in the California TTBR Hart source code report. The misuse of the Tally database was not previously reported.

Prerequisites: An attacker needs the MBB key (for forging MBBs), which can be obtained from the eCM tokens or via other means (see Issues 19.2.2, 19.1.7), or access to the Tally database.

Impact: This vulnerability allows an attacker to prevent legitimate votes from being counted.

Procedural Mitigations: The integrity of the Tally database must be ensured. Access should be strictly controlled to the county-wide key, and strict physical control and inventorying of the MBBs is required such that only MBBs from legitimate voting machines are provided to Tally.

Verification: This vulnerability was confirmed through source code analysis.

19.9.5 Rally and Tally allow a user to accept previously unrecognized certificates

Rally and Tally protect their connection with OpenSSL, which provides encryption and authentication over a network connection. This connection is set up through the use of certificates, which are digitally signed by a certificate authority trusted by both Rally and Tally. In the implementation of Rally and Tally, there is no reliance on a certificate authority; rather, each application will obtain certificates from the first connection made to them. Thereafter, if the connecting software presents a certificate other than the originally supplied, the user is prompted as to whether the new certificate should be should be accepted. This certificate is used to authenticate Rally to Tally and vice-versa. Thus, anyone who connects to Tally or Rally can pretend to be the other application, and that application will simply prompt an unsuspecting user to accept the replaced certificate.

Description: The Rally and Tally software perform no real certificate management on behalf of the user, but assume the user is competent to judge the received certificate based only on some asserted source. An improved setup would have the key of the certificate authority installed in both applications, such that they could verify that a received certificate was digitally signed by the trusted certificate authority. As part of the initial mutual authentication process, the user accepts the certificate by hitting an "OK" button in a presented interface. Thereafter, if a new certificate is presented, the same accept dialog is repeated.

The interface presented to the user includes information about the organization, city, state, and country of the certificate owner. Note that an adversary can simply reuse the same information in an original legitimate certificate, as one can create a certificate asserting any identity desired. Thus, the user has no meaningful information to use when deciding if a received certificate is authentic. This introduces opportunities for *man-in-the-middle* attacks, where the attacker is able to intercept traffic between two parties and alter the contents so that the messages the attacker can choose what messages are received by each party, or outright *masquerading* attacks, where the attacker pretends to be the other party.

This confirms Issue 32 reported in the California TTBR Hart source code report.

Prerequisites: Access to the Ethernet interface on Rally or Tally.

Impact: Allows the adversary to impersonate a Rally or Tally host, or to act as a man-in-the-middle (and observe and modify all communication between the two).

Procedural Mitigations: Develop careful controls over how and when certificates should be accepted.

Verification: This vulnerability was confirmed through source code analysis.

19.10 Hart Ballot Now Vulnerabilities

19.10.1 Ballot Now ballot counters are stored in a database

Ballot Now keeps a private counter of the number of ballots that have been processed, and a public counter of the number of ballots processed for a given election. An attacker, such as a malicious election official who has access to the database, can modify these values directly, thereby rendering the ballot allocation and process checks unusable.

Description: The state of the system is largely captured in Ballot Now in an internal database. Among the values stored are counters that record the number of ballots that have been processed by the system for a particular election. These counters may be changed by anyone able to connect to the Ballot Now database. Passwords to access the database are easily obtainable, as they are insecurely stored (as described by Issue 19.1.2).

This confirms Issue 16 reported in the California TTBR Hart source code report.

Prerequisites: Access to the database.

Impact: This attack allows the attacker to falsely cast doubt on the correctness of an election. In addition, it may be used to cover another attack, such as ballot forgery.

Procedural Mitigations: One needs to ensure the attackers cannot access the database.

Verification: This vulnerability was confirmed through source code analysis.

CHAPTER 20

HART NEW ISSUES

This chapter describes vulnerabilities we found in the Hart InterCivic election management system, beyond those discovered in the California TTB Source Code Report for Hart. Note that as discussed in Section 2.4, we were not given access to a central count optical scanner, used for processing absentee ballots with Ballot Now. This limited our ability to test the operation of Ballot Now as an input processing device during elections. In addition, due to time restrictions, we only performed a cursory overview of the Disabled Access Unit (DAU)-specific features of the eSlate. However, many of these issues have been covered in a recent academic report focused on study of the DAU.[1]

The public version of this report contains references to sections of the private report. These redacted sections contain confidential information identifying where in the source code or design the issue is manifest, or other vendor-confidential information. Note that all source code, file names, and other apparent vendor specific information in the public version have appeared in previous election study analysis reports.

20.1 Hart General Vulnerabilities

20.1.1 An MBB image can be copied and restored without any credentials.

The data on an MBB (called the MBB "image") can be removed from the card and restored using simple tools. This allows an attacker to effectively edit out certain parts of an election by periodically removing the card and saving it to its previous state. Thus, voters will have their votes erased from the MBB during an election. Unless the internal CVR logs of the devices in question are examined after the election, the modified votes may never be noticed.

Description: The MBB is a readable and writable electronic storage device similar to a hard or floppy disk. At a low level, it is simply a persistent device that can be read and written to freely by an operating system. An attacker can exploit this design by simply copying the data off the MBB (and onto, for example, a local hard drive) and restoring using commonly available tools such as the UNIX `cpio` utility. This feature was enormously helpful in our analysis; it allowed us to save the state of an election at any time and replay and experiment with it later.

This feature can be exploited by an attacker to manipulate an election. For example, assume an attacker knows that a set of voters who vote over lunch are predisposed to vote in a certain way that is undesirable

[1] Elliot Proebstel et al., 'An Analysis of the Hart Intercivic DAU eSlate'. In Proceedings of the USENIX/ACCURATE Electronic Voting Technology Workshop. August 2007.

to him. He could then copy the MBB image onto a hard at 11:30am and reinsert it back into the election device. At and reinsert back into the device. When the device boots again, the preceding 2 hours would be effectively erased from the MBB with no visible notification. This would prevent the votes over that period from being counted; in fact, there would be no indication that they had voted at all apart from the internal logs in the devices.

Further, if that same attacker has access to the VVPAT, he can simply cut/tear out the removed entries. Thus, there will be no evidence of the votes occurring other than a count of the number of voter sign-ins.

Prerequisites: The attacker needs access to the MBBs, and possibly the VVPATs.

Impact: This attack may allow an attacker to remove targeted votes from an election.

Procedural Mitigations: Any discrepancy between the voters signed into a precinct and the number of votes counted should be flagged as a potential compromise. Note that this would not correct the problem (there is no correction), but simply detect the problem potentially exists.

Verification: We verified this issue via demonstration. We were able to remove a voter's vote from being tallied by copying and restoring an MBB.

20.1.2 EMS systems make improper use of the Windows registry

The Windows registry is used to store configuration data for applications running on a windows system. Each configuration parameter is stored in a single registry entry. EMS applications use the Windows registry to store configuration parameters that can enable debugging functionality, reveal sensitive information and disable or modify the behavior of EMS applications.

Description: The Windows registry is used for storing configuration parameters used by applications. The Hart EMS applications misuse the registry in several ways.

- Debugging features can be activated through the registry.
- Confidential information is stored in the registry.
- The registry is used to control the functionality of user interfaces.

The practice of leaving debugging or "back door" features in production level software is considered dangerous. Debugging functionality left in applications can expose less tested parts of the application which may lead to the compromise or misuse. The use of registry entries to control debugging features of Ballot Now is discussed in Issue 20.7.2.

Many of the registry entries on the EMS machines can be read by any user on the machine. For this reason, any confidential information such as passwords stored in the registry will be viewable by any user on the system. The password for the Ballot Now security database private key is stored in the registry, and thus is readable by any user on the system. An attacker able to read this key can monitor and potentially modify the traffic between Ballot Now and the Ballot Now Security database.

The registry is used to store parameters affecting the availability and behavior of user interface components in Tally. Misconfiguration or tampering of these parameters can make Tally unusable or cause subtle errors in its operation, which could be overlooked by the user. The use of registry entries to configure the Tally user interface is discussed in Issue 20.6.2.

Prerequisites: In order to leverage registry related issues, an attacker will need physical access to a Hart EMS machine.

Impact: An attacker able to modify registry entries on EMS machines can print pre-voted ballots, view confidential information and modify the Tally user interface.

Procedural Mitigations: Limiting physical access to the EMS machines at all time may reduce the opportunities for an attacker to tamper with the registry. Ensuring that the EMS systems are never connected to a network may also reduce such opportunities.

Verification: We verified these vulnerabilities by enabling the Autovote functionality in Ballot Now, [2] and removing and modifying the behavior of user interface components in Tally. [3] We did not attempt to monitor or modify traffic between Ballot Now and the Ballot Now security database by decrypting the Ballot Now private key using the password stored in the registry.

20.1.3 Hart EMS passwords can be bypassed

The EMS applications, BOSS, Ballot Now, SERVO and Tally, all require a username and password to login. An attacker can bypass having to enter these credentials by modifying the security database associated with each application.

Description: Each EMS application requires users to enter a username and password before using the application. Once the user's credentials are entered, they are checked against values stored in the security database for each application. If no usernames are present in the security database, the application will prompt the user for a new administrator username and password, and use them to create a new administrator account which the user can then login with.

As described in Issue 19.1.2, an attacker can connect to the EMS databases, including the security databases, and modify their contents. Once an attacker has deleted the usernames in the security database for an EMS application, he may open the application, create a new administrator account, and login using that account.

Prerequisites: In order to exploit this vulnerability, an attacker must have physical access to any of the EMS machines, and to have extracted the database passwords from the configuration files as described in Issue 19.1.2.

Impact: An attacker that can login to the EMS applications can completely undermine any or all of the following.

- Ballot definition
- Ballot creation
- Vote tallying
- PVS maintenance

The specific actions that can be performed by an attacker with access to the EMS applications include the following.

- Adjusting the vote totals in Tally

[2] See Issue 20.7.2 for more information on this vulnerability.
[3] See Issue 20.6.2 for more information on this vulnerability.

- Adding new user logins to EMS applications
- Clearing the audit logs stored in the JBC, eSlate, eScan and MBBs with SERVO
- Printing Ballots with Ballot Now

Procedural Mitigations: Limiting physical access to the EMS machines would reduce the opportunities an attacker would have to connect to the EMS security databases. Ensuring that the EMS systems are never connected to a network would also reduce such opportunities.

Verification: We connected to the security database for each EMS application, BOSS, Ballot Now, Tally and SERVO, as described in Issue 19.1.2, and deleted the usernames. This allowed us to create new administrator accounts upon starting each application.

20.1.4 Hart EMS audit logs can be modified or erased

The EMS applications, BOSS, Ballot Now, SERVO and Tally, all maintain audit logs of the functions they have performed. The EMS audit logs are stored in the databases for each application. An unprivileged attacker may open these databases and modify the audit logs, covering any evidence of tampering with EMS applications and election data.

Description: The EMS audit logs are used to maintain an audit trail of the actions taken by EMS applications. An attacker that has tampered with the EMS applications, perhaps by circumventing the passwords as described in Issue 20.1.3, could completely remove and trace of this tampering in the EMS audit logs.

Each audit log entry contains at least the following.

- **date and time**: The date and time at which the action was logged
- **user**: The name user that performed the logged action
- **code**: A unique identifier used to identify the action performed, the code/description pairs are located in another table in the database
- **data**: Any special information recorded about the log entry, for example, if the vote totals were adjusted, the **data** field will contain the amount of the adjustment.

Once an attacker has connected to an EMS database, suspicious audit log entries can be removed in several ways. The simplest way would be to delete all audit log entries. This method will also arouse the most suspicion, as some EMS applications place a sequential **audit ID** number with each audit log entry. A more discreet method would be to delete entries by **audit ID** and replace them with entries with less suspicious **codes** for actions such as "successful login," with the same **audit ID**.

Prerequisites: In order to exploit this issue, an attacker will need access to the Hart EMS systems, and to have extracted the database passwords from the configuration files as described in Issue 19.1.2. With this access, modifications to the audit log for a single entry could be performed within seconds, while large-scale modifications could be performed in minutes.

Impact: An attacker able to modify the audit log can make it difficult or impossible to detect that an EMS application has been tampered with. This would allow an attacker to cover up having performed any and all functions of the EMS applications discussed in Issue 20.1.3.

Mitigation Strategies

Procedural Mitigations: Limiting physical access to the EMS machines may reduce the opportunities an attacker would have to connect to the EMS databases and modify the audit logs. Ensuring that the EMS systems are never connected to a network may also reduce such opportunities.

Verification: We connected to the databases for each EMS application, BOSS, Ballot Now, Tally and SERVO, as described in Issue 19.1.2, and modified the audit logs to remove specific entries. We then replaced the removed entries with less suspicious **code** fields, a method which would arouse little or no suspicion on the part of one inspecting the logs.

20.2 Hart eCM Vulnerabilities

20.2.1 eCM keys may be quietly recorded to a debug file

The EMS applications (BOSS, Ballot Now, Tally, SERVO and eCM manager) all use a Spyrus library for accessing the keys stored on the eCM tokens. This library checks a Windows registry key for a path to a debugging output file. If found, the library will write debugging information, including the eCM key or part of the eCM key to the debug file. An attacker able to set this registry entry can obtain the eCM key used in a precinct, and thus create forged MBBs.

Description: BOSS, Ballot Now, Tally and SERVO all use eCM tokens (Spyrus Rosetta USB PKCS#11 cryptographic tokens) to compute cryptographic hashes. The EMS applications use a Spyrus library to access these tokens. This library checks a "Debug / Filename" registry entry. If the registry entry is found, the library will print debugging information to this file. This file contains the eCM key as well as the key GUID. From our observations, at no point in time during the execution of the EMS applications was the user informed that the eCM key or any other information was being written to a debug file.

For each EMS application, any operation that reads the eCM key causes 16 out of 40 bytes of the key to be written to the debug file. The eCM manager "Write Module" operation, used to write the key to an eCM token causes the entire key to be written to the debug file. Because this operation is used in the creation of eCM keys, an attacker able to set the debug registry entry any time before key creation can obtain the key that will be used in the precinct before ballot definition or creation has begun.

Prerequisites: In order to exploit this issue, an attacker will need physical access to the EMS machines, and know the proper registry entry to specify the path to the debug file. This is not difficult to find, as the registry key used for the debug parameter is located in the Spyrus DLL used by the EMS applications.

Impact: An attacker that can set this registry key can obtain the eCM signing key necessary to forge MBBs.

Procedural Mitigations: Strictly limiting physical access to the machines running EMS applications may reduce the opportunities for an attacker to create the Spyrus debug registry entry or subsequently be able to retrieve the key from the machine. Ensuring that the EMS systems are never connected to a network may also reduce such opportunities.

Verification: This issue was confirmed via demonstration with the EMS applications. We set the debug registry entry, and ran each application. After running the EMS applications, we checked the contents of the debug file, and found 16 bits of the key present in the debug file. After performing the eCM Manager "Write Module" operation, we found the entire key present in the debug file. We also verified that the path to the debug registry entry was in the Spyrus DLL on the EMS machines.

20.3 Hart eScan Vulnerabilities

20.3.1 The flash memory containing the eScan executable and file system can be replaced

The eScan maintains its firmware image and file system on a compact flash card. An attacker can change this card in under two minutes, completely subverting the operation of the eScan.

Description: The eScan's firmware and file system are located on a standard Compact Flash card. To access this card, an attacker must remove the screws around the base of the perimeter of the eScan. Two of these screws are covered by tamper evident seals. Upon removing these screws, the top of the eScan may be removed. The Compact Flash card, which is located beneath an IDE cable on a daughter board, may then be removed. There is one more tamper evident seal on the Compact Flash card.

Prerequisites: In order to replace the eScan internal flash memory, an attacker will need physical access to the eScan for two minutes.

Impact: An attacker able to replace the eScan's internal flash memory can completely control the eScan, thereby controlling election results and compromising voter privacy in a precinct.

Procedural Mitigations: Limiting physical access to the eScan as much as possible can reduce the opportunities for an attacker to replace the flash memory card. Tamper-evident seals are a potential *partial* mitigation to prevent unauthorized access. For practical limitations, see Section 3.5.

Verification: We loaded the Linux operating system onto a Compact Flash card, opened the eScan, installed the card and closed the eScan in under 2 minutes. We then successfully booted the machine with the new memory card.

20.3.2 The eScan runs a telnet server

The eScan runs a telnet server, a program that readily accepts network connections. Telnet is an old and insecure protocol, and there are known vulnerabilities in widely used telnet servers. An attacker able to communicate with the eScan's telnet server may be able to subvert the eScan.

Description: The telnet server running on the eScan is the Microsoft Windows CE Telnet service. Several vulnerabilities for other Microsoft telnet servers are known[4], suggesting the telnet server on the eScan may also be vulnerable. If an attacker knew of a vulnerability in the Windows CE Telnet server, it could be possible to compromise the eScan. We cannot speculate as to the purpose of the telnet server as we could not find any reference to it in the Hart documentation we received.

Prerequisites: In order to communicate with the telnet server, an attacker must have physical access to the eScan's Ethernet port.

Impact: An attacker able to leverage a vulnerability in the telnet server running on the eScan could possibly execute arbitrary code, completely compromising the machine. This would allow for the modification of election results and the compromise of voter privacy for a precinct.

[4]Microsoft Corporation, *Microsoft Security Bulletin (MS00-0500)*. July 2000 ⟨URL: `http://www.microsoft.com/technet/security/Bulletin/MS00-050.mspx`⟩; Microsoft Corporation, *Microsoft Security Bulletin (MS02-004)*. February 2004 ⟨URL: `http://www.microsoft.com/technet/security/Bulletin/MS02-004.mspx`⟩.

Procedural Mitigations: Access to the telnet server may be limited by blocking the Ethernet port on the eScan during elections.

Verification: We connected to the telnet server over port 23 and reached a login prompt. We could not successfully login to the telnet server.

20.3.3 The eScan scanner surface can be occluded to affect ballot processing

The eScan scanner surface can be accessed by lifting the unsealed device doors. An attacker can place inks or tape over certain parts of a surface to prevent ballots from being processed or votes for targeted races from being counted. In some cases, the voter will not be able to know that his votes were ignored.

Description: The eScan scanner feeds the voter's completed ballot between two small glass scanning windows above and below the paper. Optical scanners "read" both sides of the ballot as it is fed by paper rollers. In this way, the scanner scans the ballot either face-down or face-up.

An attacker who can paint opaque inks or place tape over the glass plates can affect the way the eScan processes ballots. The tape or ink is aligned vertically with the vote "boxes" in the roller feed direction. In this way, the optical sensor is blinded from seeing the boxes passing over the occluded area. The inks/tape can be placed such that the same races can be blocked when fed in either face-up or face-down.

The eScan glass plates are readily accessible. A door on the right side of the scanner can easily be opened and inks or tape aligned and applied in a few seconds. This attack may be difficult for an voter to mount as the eScan is likely to be attended by a poll worker. In a more subtle attack, a voter may be able to place the inks on a ballot, and then feed it through the scanner. The inks would then smear across the surface resulting in the same "blinding" of the optical sensors. This attack would likely be undetectable until the next voter votes.

When the ballot is scanned using black or reflective tape (or ink), the scanner indicates an over-vote of the first vote in the vertical column–the over-vote can be accepted by the poll official pressing the "over-vote" red-button on the back of the computer. However, instead of the one contest being over-voted *all* contests vertically aligned on the ballot are ignored as over-votes. The discarding of votes for the latter races occurs without voter notification. When the ballot is scanned with white or red tape/ink, is simply rejected as not being scanned properly.

Note that there is an option programmed into the MBB that allows election officials to disable any notification of over- or under-vote. Thus, in those precincts where this features was enabled, the votes would be uncounted silently; the voters would receive no indications that that their vote was not counted.

If permanent inks were used, the eScan would be unusable until its glass surface was replaced. This could seriously inhibit the processing of votes within a precinct.

Prerequisites: Access to the eScan, or in the voter attack, access to a ballot.

Impact: This would prevent votes for a given race from being counted, or disable the eScan entirely.

Procedural Mitigations: The door to scanning surface should be adequately sealed.

Verification: This issue was confirmed via red-teaming as described above.

20.3.4 The eScan ballot box collecting votes allows vote order to be reconstructed

The eScan optical scanner is placed in an "eScan Ballot Box", a large plastic tub that collects the paper ballot fed through the scanner. The ballots drop in one by one and there are no mechanisms in place to change the order in which they fall. Thus, specific voters may be targeted by noting the order in which they vote and examining the pile of ballots at the close of the election.

Description: During election day, an eScan optical scan unit is placed in a special housing called an eScan Ballot Box. This is a wheeled plastic tub that has a lock to keep the eScan in place and a side door that can be used to retrieve the cast paper ballots after an election. The door is locked on election day and when the eScan is in place, when a ballot is fed through it, the ballot subsequently drops through a small opening into the tub. The path of the paper when it is dropped into the tub generally causes it to drop into a pile that may be subsequently retrieved at the close of the election.

We examined the tub and discovered that there are no mechanisms within it to inhibit or otherwise change the way that ballots were dropped into the pile, or to (mechanically or otherwise) re-order or shuffle the ballots in any way. As a result, at the close of an election, many of the ballots will be in the pile in the order in which they were entered into the eScan, i.e., the first ballot cast will be at the bottom of the pile. While it is possible that some ballots may end up counted randomly, it appears that the majority of ballots will maintain their cast order.

We cast 10 ballots through the eScan machine that we numbered in order; upon checking the tub, we examined the pile of ballots and found that the order had been maintained. A watchful adversary who had access to the ballots after they had been cast would be able to report, with high probability, the way that a given voter had cast their ballot if they knew the order in which that vote had been cast.

Prerequisites: An attacker would need to watch the order in which votes were cast at the eScan machine, then be able to access the pile of ballots at the end of the election. A malicious poll worker, for example, could determine the order of a cast ballot and make note of the way in which that voter had voted.

Impact: This allows a compromise in voter privacy, as an attacker could determine the way a vote was cast. This attack could be used for the purposes of vote-buying or coercion.

Procedural Mitigations: The election administrator should shuffle the pile of ballots in the eScan ballot box at the close of the election. This will have limited effect if the administrator or any workers observing the pre-shuffled state of the ballots is corrupt.

Verification: This issue was confirmed via red-teaming as described above.

20.3.5 The eScan ballot box is vulnerable to attacks that may destroy ballots

As described in issue 20.3.4 the "eScan Ballot Box" is a large plastic tub that collects the paper ballot fed through the eScan optical scan unit. The ballot box has an "emergency ballot slot" for use if power to the eScan goes out; voters can cast their ballots through this slot into a small, removable tub. This tub has a hole in it, allowing an attacker to pour corrosive liquid into the unit through the emergency slot and having it leak onto the main pile of ballots, which can be dissolved or severely damaged.

Description: The emergency ballot slot on the eScan Ballot Box is meant to be used if the eScan becomes inaccessible, such as in the event of a power outage to the unit. A tub is installed inside the ballot box in order to collect ballots cast this way, which are meant to be counted and tallied at election headquarters, since the

eScan is incapable of tallying them. The door to access both the pile of ballots that fall into the box from the eScan and the tub containing ballots cast through the emergency slot is locked. A hole approximately one inch in diameter is in the tub on the opposite side from where the emergency ballot slot on the ballot box is located. This hole is presumably used to easily handle the tub so that it may be easily removed from the ballot box.

The tub is located above where the ballots normally cast through an eScan would collect. By partially filling the tub with a quantity of corrosive liquid such as acid-based drain cleaner, with a high percentage of sulfuric acid, the liquid would fall through the hole and onto the paper ballots below, dissolving them. Additionally, the unit is top-heavy because of the weight of the eScan on top of the ballot box. If someone was to fall into the unit can cause it to tip a few degrees, this would be sufficient for even a small quantity of a corrosive agent to slide down into the hole and subsequently onto the ballots beneath. The emergency ballot slot is approximately 8.5 inches wide by 1 inch high, which is more than sufficient for an attacker to surreptitiously pour a noxious substance into the tub. The gate to this ballot slot can be closed when emergency balloting is not in used, but there is enough of a gap that flexible tubing may be inserted through for liquid to be poured in.

Prerequisites: An attacker needs to have access to a corrosive liquid such as acid-based drain cleaner. They would need to be able to pour the liquid into the tub, which could be surreptitiously performed while they are voting at the eScan machine.

Impact: This attack could cause the paper ballots to be severely damaged or destroyed outright. Because the paper ballot is the official ballot of the election, destroyed ballots inside the voting machine could cast the results in significant doubt.

Procedural Mitigations: The emergency ballot slot should be sealed shut. Ballots should be collected in another fashion if there are power or technical issues with the eSlate unit.

Verification: We poured liquid into the collection tub and simulated falling into the machine to knock it askew and the liquid into the hole. We did not test the attack with sulfuric acid.

20.3.6 The eScan may be modified to allow casting of duplicate ballots

The configuration file for the eScan may be modified to allow the eScan to cast duplicate ballots. We confirmed in Section 19.4.1 that the eScan file is accessible over an Ethernet port and may be both downloaded and uploaded. A configuration parameter in this file allows duplicate ballots to be scanned, overriding controls in the eScan that should otherwise prevent this from occurring.

Description: The configuration file is common to all eScan devices. It is a text file that is found on the eScan, accessible by connecting to the eScan's Ethernet port with the correct IP address. [5] An option in the configuration file to allow duplicate ballots to be accepted is commented out by default. Setting this option in the configuration file and uploading it to the eScan via the Ethernet interface will enable the counting of duplicate ballots. These can be photocopies of a single ballot, or the same ballot used multiple times, as described in Issue 20.3.9. The only indication of a duplicate ballot having been scanned is the fact that the serial number of the ballot is logged in the eScan's audit log. Unless the log was closely scrutinized, however, it would be difficult to notice that multiple ballots with the same serial number were cast, particularly if they were interspersed amongst many legitimate votes.

It is not clear why an option to allow duplicate ballots is available on production eScan units. This may be

[5] More information about connecting to the eScan may be found in Issue 19.4.1.

a testing feature that is not meant to be enabled; however, the functionality is easily accessible.

Prerequisites: An attacker needs to have access to the Ethernet port of the eScan and a computing device, such as a Palm handheld, to connect to the port and upload a new eScan file. The attacker would then need access to a single ballot which may be arbitrarily duplicated.

Impact: This attack allows for "ballot stuffing" with an arbitrary number of duplicate ballots. If combined with attacks to erase or modify the audit logs of the eScan after the election, as detailed in Chapter 21, these ballots will not be detected except by manual recount, and only if the examiners closely note the serial number of the ballots.

Procedural Mitigations: The configuration files of all machines should be be downloaded and examined before an election. The Ethernet port on eScan devices should be sealed to prevent access during an election.

Verification: We retrieved the configuration file from the eScan unit, modified the file to allow duplicates, and uploaded it back to the eScan. We then scanned a ballot into the eScan repeatedly and it worked every time. We also made photocopies of the original ballot and scanned those in. The eScan accepted every ballot.

20.3.7 The eScan has an open interface allowing erasure of vote and audit log records

The eScan is accessible through its Ethernet port, as described in Issue 19.4.1. Using a device such as a smartphone or a handheld computing device (e.g., a Palm device), an attacker can issue commands to the eScan that include resetting it and clearing its internal records and logs.

Description: Before an election occurs, all voting equipment is initialized using the SERVO application. Part of the initialization process includes clearing the cast vote records (CVRs) and audit logs from the voting equipment, notably the eSlate, JBC, and eScan devices. The eScan is managed by SERVO through its Ethernet interface. Performing the erase action does not require the use of an eCM cryptographic token.

With a computing device such as a Palm handheld computer and a cable that will interface to the Ethernet port on the eScan, an attacker can mimic the actions of SERVO to the eScan during an election, causing the CVR and audit log records to be erased from the machine. There is no authentication performed to determine the machine interfacing with the eScan. So, these commands will be executed by the eScan regardless of the device that issued them. Any voting that has occurred on the machine to this point will be erased from the eScan's internal logs as well as from the MBB.

Prerequisites: An attacker needs to have access to the Ethernet port of the eScan and a computing device, such as a Palm handheld, to connect to the port. If an attacker is left unsupervised for only 30 seconds or less at the polling place, this attack would be possible. A poll worker could easily mount the attack if they have access to the eScan for approximately 30 seconds.

Impact: Any electronic evidence of voting having occurred on the eScan will be removed from both the eScan's internal logs and the MBB with this attack.

Procedural Mitigations: The Ethernet port on the eScan should be sealed on election day, as there is no need for it to be accessible by any device or application that is not SERVO. Poll workers should never leave the eScan unguarded. However, if the attack is carried out by an insider, this mitigation will not work.

Verification: This vulnerability was confirmed through source code analysis. We also wrote a program to allow us to reset the eScan from a laptop computer connected through the eScan's Ethernet interface. With