Security Evaluation of ES&S Voting Machines and Election Management System

Adam Aviv Pavol Černý Sandy Clark Eric Cronin Gaurav Shah Micah Sherr Matt Blaze
{aviv,cernyp,saender,ecronin,gauravsh,msherr,blaze}@cis.upenn.edu
Department of Computer and Information Science
University of Pennsylvania

Abstract

This paper summarizes a security analysis of the DRE and optical scan voting systems manufactured by Election Systems and Software (ES&S), as used in Ohio (and many other jurisdictions inside and outside the US). We found numerous exploitable vulnerabilities in nearly every component of the ES&S system. These vulnerabilities enable attacks that could alter or forge precinct results, install corrupt firmware, and erase audit records. Our analysis focused on architectural issues in which the *interactions* between various software and hardware modules leads to systemic vulnerabilities that do not appear to be easily countered with election procedures or software updates. Despite a highly compressed schedule (ten weeks) during which we audited hundreds of thousands of lines of source code (much of which runs on custom hardware), we discovered numerous security flaws in the ES&S system that had escaped the notice of the certification authorities. We discuss our approach to the audit, which was part of *Project EVEREST,* commissioned by Ohio Secretary of State Jennifer Brunner.

1 Introduction

In response to concerns about the security and reliability of electronic voting systems, Ohio Secretary of State Jennifer Brunner initiated the "Evaluation & Validation of Election-Related Equipment, Standards and Testing (EVEREST)" [18] study in October 2007. Similar in scope to the California Secretary of State's top-to-bottom review of electronic voting systems [8, 7, 2, 15, 3, 25, 6, 23], the EVEREST project aimed to identity potential weaknesses in Ohio's voting systems – specifically, systems developed by Election Systems and Software (ES&S), Hart InterCivic (Hart), and Premier Election Solutions (formerly Diebold).

In this paper, we summarize the results of our audit of the ES&S voting system for the Ohio EVEREST study. EVEREST was the first major study of ES&S voting systems, despite the system's popularity (ES&S claims to be the world's largest e-voting systems vendor [1], supporting more than 67 million voter registrations with 97,000 touchscreen voting machines installed in 20 states and 30,000 optical ballot readers present in 43 states [4]), and only the second comprehensive study that examined all components – from backend registration systems to frontend ballot casting – of any electronic voting system. In a ten week period, our seven-member team was tasked with analyzing the nearly 670,000 lines of source code that comprise the ES&S system, encompassing twelve programming languages and five hardware platforms[1].

Currently deployed electronic voting systems are inarguably complex (often involving hundreds of thousands of lines of code), and are subject to numerous potential attacks, including denial-of-service, alteration or forgery of precinct counts, compromise of official tallies, machine firmware, ballot displays, audit/recount data, and ballot secrecy violations, among others. Since an exhaustive exploration of the entire system was infeasible given our ten week mandate (to say nothing of the huge code base), our approach focused on examining the modules that we believed to be most likely problematic – in particular, those dealing with pseudorandom number generators, cryptography, memory management, and input processing.

Despite our admittedly incomplete analysis, we identified numerous critical vulnerabilities in nearly every component of the ES&S system. We describe serious practical and undetectable attacks that could be carried out by pollworkers, and in many cases, individual voters. (As a noteworthy example, a voter can surreptitiously delete all vote

[1] During the EVEREST study, we examined the specific ES&S hardware and software versions (see appendix for specific version numbers) that are certified for use in Ohio. Although other states may use different ES&S components, we believe that most of our findings, particularly the systemic issues identified in Section 4, are likely applicable to other versions of the system as well.

data and audit logs stored on a touchscreen voting machine using a Palm handheld device and a small magnet.) Such attacks could alter or forge precinct results, install corrupt firmware, and erase audit records. Of particular concern, several attacks could spread virally, propagating between voting machines and backend systems, and *vice versa*.

This paper focuses on architectural issues of the ES&S voting system. We describe vulnerabilities in each of the ES&S components, and show how the *interaction* of the various software and hardware modules leads to systemic vulnerabilities that do not appear to be easily countered with election procedures or software updates. In particular, we saw evidence of four fundamental and pervasive deficiencies that gave rise to many of the most serious security flaws.

2 Related Work

The security and reliability of electronic voting systems have long been a matter of concern. In 2003 Kohno et al. [17] published a study in which they listed numerous security flaws found in a leaked version of the Diebold BallotStation source code. Since that time, several states, most notably Maryland [20], Ohio [14], California [24], and Florida [11, 10, 25] have authorized independent studies of the voting systems used in their elections. Additionally, several independent researchers have released reports analyzing voting machine hardware and software security [9, 12, 13].

All of these studies report numerous security vulnerabilities in the machines or software they reviewed, and with one exception [8], all of the studies focus their attention on an individual model of a voting machine (predominantly Diebold Accuvote DREs), or a single aspect of a voting system, such as the backend database or the firmware. The first documented in depth analysis of an entire state's voting system was performed by the California Top-to-Bottom Review [8]. However, ES&S systems were not evaluated in that study.

A few studies have examined the ES&S products. The review commissioned by Florida Department of State [25] looked exclusively at the iVotronic firmware version 8.0.1.2. It did not examine any of the other ES&S software or hardware. The review commissioned by the California Office of the Secretary of State [5], published after the EVEREST study, looked at the source code used by ES&S machines, but did not analyze the hardware. Additionally, our analysis was conducted in parallel with an analysis by an independent security assessment company MicroSolved, Inc., with SysTest Labs [19]. While this concurrent study found many vulnerabilities, they did not have access to the source code.

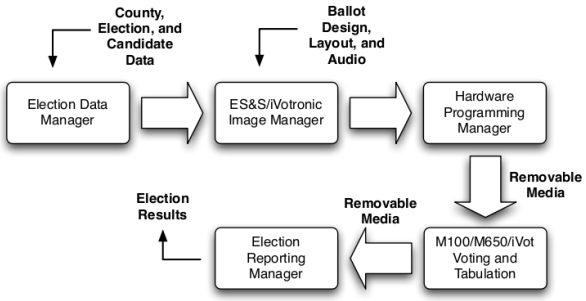

Figure 1: The high level overview of the ES&S Unity voting system with user input to each stage of the election.

Figure 2: An ES&S "Personalized Electronic Ballot" (PEB) used to transport ballots and results between the iVotronic and Unity.

To our knowledge, our report provides the first comprehensive, in depth analysis of the entire ES&S voting system.

3 ES&S System Overview

In this section we will describe how the individual components of the ES&S system inter-operate to conduct an election. We will begin by providing a high level overview, and then discuss relevant components in more detail.

3.1 Architectural High Level Overview

The election management system (EMS) from Election Systems & Software (ES&S) prepares, collects, and tabulates elections using either paper ballots or touchscreen terminals (or a combination of both). It contains software for managing all stages of an election, as well as special hardware interfaces for configuring and retrieving results from the election devices. The high level architecture of the ES&S system is shown in Figure 1.

Unity is the election management software suite. It is comprised of a number of individual programs which interact with one another through shared data files (collectively referred to as the "*database*"). *Election Data Manager* is used to initialize the database with jurisdiction, voter, and candidate information. *ES&S Image Manager* and *iVotronic Image Manager* are used to design the appearance of paper and touchscreen ballots. The *Election Reporting Manager* is used to collect and tally election results, and the *Audit Manager* is used to verify election results using audit data. All interaction between Unity and voting hardware is controlled by the *Hardware Programming Manager* program.

The *iVotronic* is a touchscreen direct recording electronic voting terminal (DRE). There are two distinct types of iVotronic terminals, distinguished by colored inserts along the sides: red *supervisor terminals* and blue *voter terminals*. Both types of iVotronic terminals are activated using special hardware tokens called *Personalized Electronic Ballots (PEBs)*, which are also used to store ballot definitions and election results (see Figure 2). PEBs are typically programmed via a supervisor terminal at the start of an election, and read using either a supervisor terminal or a dedicated *PEB Reader* connected to the machine running the Election Reporting Manager at the end of an election. A PEB can be used in multiple iVotronics as long as they are qualified for the same election and polling place.

The voter iVotronics also use *Compact Flash* cards to store large ballots, audio ballots, and election result audit files. A printer which provides a voter-verified paper audit trail, known as the *Real-Time Audit Log* is connected to the iVotronic.

The *Model 100* is a machine for scanning and validating/tallying paper ballots at a polling location. The Model 100 uses *PCMCIA Memory Cards* to hold ballot definitions and tallies.

The *Model 650* is a machine for batch scanning and tallying paper ballots at a central election office. The Model 650 uses *Zip Disks* to hold ballot definitions and election tallies.

3.1.1 Unity

Unity is the Windows-based software suite for managing elections. It contains tools for creating and managing election databases (Election Data Manager), designing the appearance of ballots (ES&S and iVotronic Image Managers), tabulating and reporting results (Election Reporting Manager). Additionally, there is a tool to audit the use of the other components of Unity (Audit Manager), and a tool for abstracting programming and communicating with the various hardware components used by several Unity components (Hardware Programming Manager). The various components of Unity communicate with each other indirectly through common files stored on the Windows filesystem.

3.1.2 Hardware Components

In this subsection we will review the different hardware components of the ES&S voting system. These include the DRE iVotronic interfaces, the M100, and the M650. In addition, some of the media interfaces used by the hardware components will be discussed in more detail.

The **iVotronic** is a touchscreen DRE based on an Intel 386 processor with 1 MB of SRAM. There are four internal flash memory devices. There are two serial ports for input/output on the iVotronic, one connected to a standard DB9 serial port at the top of the iVotronic, and the other to an infrared transceiver. The external serial port is used to connect to the RTAL printer or a communications pack to report from the field, or to a computer running Unity HPM or ERM in the central election office. The infrared serial port is used to communicate with the Personalized Electronic Ballot hardware tokens. The left side of the iVotronic case has a molded socket to hold a PEB allowing IR communication as well as activation of the iVotronic power switch through a magnetic reed switch. A Compact Flash slot is also located next to the printer serial port. There are two types of iVotronic terminals used in elections: red "Supervisor iVotronics" are used by poll workers or elections officials to administer the election, and blue "Voter iVotronics" are used by voters to cast their votes.

The red **iVotronic Supervisor Terminal** is used to manage PEBs and the contents of their flash memory before, during and after elections. It plays a far more significant role in the Voter Activated Voting mode (which is not used in Ohio elections, thus not studied in our report), where at least one Supervisor Terminal must be at every polling place. In the Poll Worker Activated Voting mode used in Ohio, a Master PEB for each polling location is created from HPM using a Supervisor Terminal connected via null modem cable. Afterward, each Master PEB is then cloned several times using a Supervisor Terminal (standalone from Unity) in order to produce the Supervisor PEBs needed while the polls are open. At this point, the supervisor terminal is no longer required for opening, closing, or tallying the election.

The blue **iVotronic Voter Terminal** is the iVotronic used by voters to cast their ballot. When a qualified Supervisor PEB is inserted the iVotronic prompts the poll worker to select the correct ballot to be voted on. If a Supervisor PEB is inserted while holding the "Vote" but-

ton above the touchscreen, a service menu appears. This menu allows various settings of the terminal to be adjusted, and also provides the interface for opening and closing the polls. While in the service menu, actions performed are logged to the RTAL printer.

The **Real-Time Audit Log Printer (RTAL)** is a continuous feed thermal printer that provides the function of VVPAT (Voter Verifiable Paper Audit Trail) on an iVotronic machines. It is connected to the iVotronic by a standard 9-pin RS232 serial cable, and mounted behind a Plexiglas window next to the iVotronic.

The **Personalized Electronic Ballot (PEB)** is a palm-sized device containing a PIC micro-controller, 2MB of flash storage, a bi-directional infrared (IR) transceiver, and battery. The PEB is activated by a magnetic reed switch, and contains a magnet to activate the corresponding reed switch in the iVotronic PEB socket. The micro-controller firmware implements the passive half of a very simple command/response protocol between the PEB and host over the PEB IR port using IrDA SIR (Serial Infrared). The primary operation is reading and writing 128 byte blocks of the PEB's flash memory, or verifying the integrity of blocks using a cyclic redundancy check (CRC) stored with each block. The last memory block of the PEB is known as the Election Qualification Code (EQC) block, and serves to authenticate the PEB to the iVotronic. PEBs, along with the compact flash cards can be used to store vote tally information for an election.

The **Model 100 (M100)** is a voter-operated optical scan ballot counter intended for use at the polling location. It is mounted on top of a secure ballot box which holds the accepted (and counted) ballots and provides physical security for the M100. The M100 provided in our study used one set of identically keyed locks for physical protection of the M100 and ballots, and a second differently keyed lock for selecting the mode of the M100. Unity (via the HPM) and the M100 use specially formatted PCMCIA SRAM flash storage cards for all communications. The cards are formatted so that a small header can be loaded into the M100's RAM which contains pointers into the SRAM for ballot definitions and results counters. The same header information also informs the M100 if this card is formatted for a firmware upgrade.

The **Model 650 (M650)** is a centralized high-speed optical ballot counter intended for use at a central elections office. It scans batches of ballots, possibly from multiple precincts, and tabulates results to be transferred to Unity. Additionally, an Iomega Zip 100 drive is used to transfer ballot definitions and perform firmware updates to the M650, and carry results from the M650 to Unity. The M650 uses FAT32 formatted 100MB Zip disks to load ballot configurations and store tallies of counted ballots. The files on the disk are copied by the Hardware Programming Manager. In Windows, these disks are mounted to the desktop and accessible to any Windows application with no special libraries.

The hardware and software system components are described in detail in Chapter 5 of [18].

4 Systemic and Architectural Issues

There are fundamental security deficiencies throughout the ES&S Unity EMS, iVotronic DRE and M100 optical scanner software and hardware. Virtually every mechanism for assuring the integrity of precinct results and for protecting the back-end tallying system can be circumvented. Election results can be tampered within the ES&S system by exploiting any of a number of different vulnerabilities that were discovered. The normal access provided to individual precinct poll workers (and in some cases to voters themselves) is sufficient to conduct attacks that alter county-wide election results and that, in some cases, cannot be detected or recovered from through audits or recounts.

Perhaps more importantly, we show how the *interaction* of the various software and hardware modules leads to systemic vulnerabilities that can spread throughout the system. There is a strong potential for practical attacks that propagate "virally" from the field back to the county election management system. That is, a single circumvented piece of precinct hardware (such as a memory card returned from a precinct for vote tallying) can effectively "take over" the county-wide back-end tally system, alter county-wide results reported in the current election, and then corrupt the installed firmware of additional precinct hardware in subsequent elections. The broad scope of such attacks provides great leverage to the adversary and can be extraordinarily difficult to detect, trace, or recover from. Different possibilities of how the firmware of each component can be altered by inputs from other components are described in Section 5.

Both the DRE (iVotronic) and the precinct-based optical scan (M100) systems are subject to many exploitable vulnerabilities. However, the DRE system provides more vectors for attacks that cannot be recovered from through manual recounts. While there are many specific errors and weaknesses in various parts of the ES&S software (and which are detailed in our earlier public report[18]), our focus is on systemic weaknesses throughout the system's overall design and implementation. Hence, the following is just a partial sample of the vulnerabilities we discovered. These weaknesses render the system as a whole difficult to secure in practice. We identify four fundamental,

Figure 3: A PEB emulator running on a Palm device simulates an initialization PEB during an open election, resetting all iVotronic passwords.

Figure 4: Unhindered access to printer connection allows disabling the iVotronic VVPAT audit log as well as the ability to print unauthorized data to the audit log.

pervasive deficiencies that give rise to the most serious vulnerabilities we found: ineffective access control, critical errors in input processing, ineffective protection of firmware and software and, ineffective cryptography and data authentication.

4.1 Ineffective Access Control

The firmware and configuration of the ES&S precinct hardware can be easily tampered with in the field. Virtually every piece of critical data at a precinct – including precinct vote tallies, equipment configuration and equipment firmware – can be compromised through exposed interfaces, without knowledge of passwords and without the use of any specialized proprietary hardware.

4.1.1 iVotronic passwords and PEB-based access controls

Access to the iVotronic DRE configuration is protected by several hardware and password mechanisms, all of which can be defeated through apparently routine poll worker (and in some cases voter) access.

The primary mechanisms for preparing iVotronic DREs employ the *Personalized Electronic Ballot (PEB)* interface. As described in Section 3.1.2, a PEB is a small module that communicates with the iVotronic via a magnetically switched infrared bidirectional data interface on the terminal. PEBs are used as external memory devices that communicate through a simple protocol that allows the iVotronic to read and write memory blocks stored in the PEB. Access to PEB memory is not protected by encryption or passwords, although some of the data stored on them is encrypted (see Section 4.4). PEBs themselves are proprietary devices but they employ *IrDA*, which is a widely-used infrared communication standard.

Some of the PEB's functions are intended to be performed at the county headquarters (e.g., loading ballot definitions and basic configurations), while others are performed by poll workers (e.g., opening the terminals at the beginning of the day, enabling a voter to use a particular ballot, closing the terminal and collecting vote totals).

In spite of the proprietary nature of the "official" PEB, it is relatively simple to emulate a PEB to an iVotronic or to read or alter the contents of a PEB using only inexpensive and commercially available IrDA-based computing devices (such as Palm Pilot PDAs and various mobile telephones).

Most of the administrative and poll worker functions of the iVotronic (e.g., pre-election ballot loading, enabling voting, etc) require the insertion of a properly configured "supervisor" PEB and, in some cases, the entry of a password on the terminal touchscreen. However, it is possible to defeat both of these security mechanisms. This makes practical several possible attacks at polling stations.

Unauthorized screen calibration and configuration One of the simplest, and yet most important, configuration parameters of the iVotronic is the calibration of its touchscreen. Calibration (which is done through the screen itself) affects how voters' tactile input maps to different locations on the screen. If performed incorrectly or deliberately altered, voter choices might not be correctly recorded. Calibration can be performed, for example, in a manner that allows most input to behave normally, but that denies access to specific screen regions corresponding to a candidate selection.

Access to the screen calibration function of the iV-

otronic terminal requires the use of a supervisor PEB during power-up (e.g., after voting or at idle times). No password is required. Any supervisor PEB is sufficient for this purpose, even one not specifically configured for the correct precinct or obtained from some other jurisdiction (e.g., through secondary markets such as eBay). A homemade PEB emulator (e.g., a specially programmed Palm Pilot and a small magnet) is also sufficient. The procedure requires about a minute and is, from a distance, largely indistinguishable from normal voter behavior[2].

While a maliciously calibrated terminal may be noticed by voters and can, in principle, be corrected in the field, the attack is extremely simple for a poll worker (or other person with access to a PEB) to carry out and practical even without a PEB, and so may represent a serious practical threat. We note that iVotronic behavior consistent with such attacks has been reported in various jurisdictions during actual elections [21].

Undocumented PEB features can bypass password checks Many of the more sensitive iVotronic administrative functions (closing the polls, clearing the terminal, etc) require the entry of passwords in addition to the insertion of a supervisor PEB. However, there is a special *Quality Assurance (QA)* PEB type recognized by the iVotronic that behaves essentially as a supervisor PEB but that, when used, does not require the entry of any passwords. This PEB type does not appear to have been documented in any of the ES&S manuals or training materials provided to us during our review[3].

This undocumented PEB feature can be used to neutralize the security of any iVotronic administrative features that depend on passwords. Anyone with such a PEB (whether emulated or acquired) effectively has a backdoor that bypasses this basic security check. QA PEBs are no more difficult to emulate than regular supervisor PEBs; the only difference being a single character changed in the communication protocol.

Note that while the QA PEB bypasses password checks, there is another iVotronic security feature required for access to some (but not all) administrative functions, For these functions, a PEB must be configured with the correct *Election Qualification Code (EQC)* (a 32 bit random number assigned for each election). However, as noted in the next section, precinct poll workers (and others with brief access to the poll worker equipment) can easily extract this code from the precinct's supervisor PEB using a palmtop computer.

Note that even without the EQC, however, an attacker (who needs no more access than that provided to a normal voter) with a QA PEB (or an emulated QA PEB) can do a great deal of harm to an iVotronic terminal. For example, the EQC is not required for the "clear-and-test" routine on an iVotronic terminal, which erases all stored votes and renders the terminal useless for the rest of the election day.

Unauthorized PEB copying and alteration Anyone with physical access (or close proximity) to PEBs can easily extract or alter their memory. This requires only a small magnet and a conventional IrDA-based palmtop computer because PEBs themselves enforce no passwords or access control features[4].

The ease of reading and altering PEB memory facilitates a number of powerful attacks against a precinct's results and even against county-wide results. An attacker who extracts the correct EQC, cryptographic key, and ballot definition can perform any election function on a corresponding iVotronic terminal. An attacker who has access to a precinct's main PEB when the polls are being closed can alter the precinct's reported vote tallies, and can inject code that takes control over the county-wide back-end system (and that thus affects the results reported for all of a county's precincts).

Individual precinct poll workers have many duties that involve handling PEBs throughout the election day (whenever a voter votes, for example), and so are in a natural position to carry out attacks that involve altering or reading PEB memory without engaging in suspicious activity.

4.1.2 Physical security, locks and seals

Many aspects of the ES&S system's security as a whole depend on the integrity of the interfaces and removable media associated with precinct equipment. Some of these are protected by software security (e.g., access passwords, encryption, etc); potential attacks against such mechanisms are discussed elsewhere (see Sections 4.1 and 4.4). Many interfaces and media are also protected (partly or entirely) by physical mechanisms: locks, seals, and procedures.

Several features of the iVotronic's physical security are especially problematic. A primary mechanism for logging events (including those potentially associated with an attack) is the RTAL printer[5]. However, the cable connect-

[2]Further details about this vulnerability can be found in Sections 7.2.8 and 7.2.13 of [18].

[3]Further details about this vulnerability can be found in Sections 7.2.10, 7.2.11 and 7.2.12 of [18].

[4]Further details about this vulnerability can be found in Sections 7.2.2, 7.2.3 and 7.2.4 of [18].

[5]Further details about this vulnerability can be found in Section 7.2.14 of [18].

ing the printer is readily accessible to the voter and can be removed easily and without tools or overtly suspicious activity (see Figure 4). Using the unprotected printer interface, the attacker can also print arbitrary messages including VVPAT ballots. In addition, it is important to note that the PEB interface is exposed and facilitates the attacks noted above.

The mechanical locks supplied with all of the ES&S precinct equipment were uniformly of very low-security designs that can be easily picked or otherwise bypassed. Many locks use keys that are apparently identical in equipment shipped to different customers, and so would provide little security even if the locks were improved[6].

More importantly, all but most sophisticated commercially-available tamper seals are often surprisingly easily to defeat [16]. Even if effective at revealing tampering, seals are inherently limited in the protection they provide. As noted in previous studies [6], seals do not *prevent* tampering; at best they can *detect* it. But in an election, even reliable detection of tampering may be unsatisfying, since if a seal is found to be broken once polling has started, it is unclear what should be done. If the compromised equipment is used, fraudulent votes may be counted. If it is not used, previously cast legitimate votes may be lost (making breaking a seal a simple way for an attacker to destroy votes).

4.2 Critical Errors in Input Processing

We identified two critical components of the ES&S system which suffer from exploitable errors in functions that process input over their external interfaces. Both the Unity and the iVotronic terminal have *buffer overflows*, that allow an attacker who can provide input (e.g., on a PEB or memory card) to effectively take control over the system.

We found numerous buffer overflows throughout the ES&S system. Several of these buffer overflows have extremely serious practical security implications. An attacker who can present input using an iVotronic PEB or an M100 memory card can take control over the results reported by the entire county election system.

Most seriously, the nature of these vulnerabilities means that there are few barriers to obtaining the access required to exploit them. In the case of the iVotronic system, voter access to the terminal is sufficient. In the case of the Unity system, brief access to any iVotronic or M100 optical scan results media returned back to the county for processing is sufficient.

[6]Further details about this vulnerability can be found in Sections 7.3.4 and 7.3.5 of [18].

4.2.1 Unity

The Unity election management system processes all precinct results and produces the tally reports that, in most cases, constitute the official tallies in races. After polls are closed, precinct-counted ballot results are received into Unity through several different media, including iVotronic PEBs, iVotronic CF cards, and M100 PCMCIA memory cards.

While Unity appears to correctly process properly-formatted results from such media, buffer overflows in Unity allow a maliciously altered input to execute arbitrary code on the computer on which Unity runs.

Attack via a PEB A stack-based buffer overflow in Unity can be exploited when election, pre-election, or testing results are processed. An attacker can exploit this vulnerability by creating a specially-crafted PEB that will allow arbitrary code execution. As a result, an attacker has the ability to gain full control of the machine running the Unity server[7].

Attack via a M100 PCMCIA memory card A specially-crafted PCMCIA memory card can take advantage of the underlying assumptions by Unity on the maximal number of precincts reported per card to exploit a buffer overflow and take full control of the Unity server[8].

Corrupting county-wide results PCMCIA cards returning results from precincts are not checked by Unity to see if they correspond to cards that were actually configured for the M100 scanners being used at the polling location. A poll worker can thus take a card with election results and insert additional results for a precinct for which the card was not configured for. A malicious poll worker can therefore not only modify the results for his or her own precinct, he or she can influence the results for other precincts as well. A careful and attentive operator using Unity can catch this attack, in case he or she has a list of cards (their serial numbers) and the number of precincts that should be reported for each card.

Note that because these vulnerabilities affect the central counting system, a corrupted media attack conducted from *any single* precinct can corrupt results for the entire county[9].

[7]Further details about this vulnerability can be found in Section 7.1.1 of [18].

[8]Further details about this vulnerability can be found in Section 7.1.2 of [18].

[9]Further details about this vulnerability can be found in Sections 7.1.3 of [18].

4.2.2 iVotronic

The iVotronic terminal firmware has several exploitable buffer overflow errors in its PEB input processing functions. These buffer overflows allow a PEB containing carefully-structured data (or an emulated PEB) to take control over the terminal[10]. The implications of attacks against iVotronics are discussed in Section 4.3.

It is straightforward to exploit the iVotronic buffer overflows in several different ways (by emulation of a QA or supervisor PEB or by writing data to a precinct's supervisor PEB) at various times (while opening polls and during the polling day), and with various degrees of access (as a poll worker or as a voter). The exposed nature of the PEB port and the many different scenarios under which it can be exploited make attacks against the iVotronic very difficult to effectively guard against under operational election conditions.

4.3 Ineffectively Protected Software and Firmware

The integrity of election results depends heavily on the integrity of the software and firmware that runs the central election management system and the precinct hardware. The consequences of any attack that alters, replaces or otherwise compromises this software or firmware are sweeping and often impossible to recover from. The security features that protect election software and firmware from unauthorized tampering are therefore among the most critically important safeguards in the system as a whole.

We found exploitable vulnerabilities that allow an attacker to replace or alter the firmware and software of virtually every component of the ES&S system, either by circumventing access controls or by triggering software errors.

4.3.1 iVotronic firmware

The iVotronic terminal is based on an Intel 80386 embedded computer processor controlled by firmware stored on an internal flash memory chip. The firmware is designed to be field-updated through an administrative menu function, with new firmware loaded though the terminal's CF card interface. Four security mechanisms are intended to protect against unauthorized firmware loading:

- Access to the firmware update menu function requires a supervisor (or QA) PEB.
- A 6-8 character password is required to enable firmware update.
- The firmware is loaded through the CF card interface, which can be protected by a sealed sliding cover.
- The firmware update function is disabled while the polls are open.

Unfortunately, these mechanisms are ineffective. There are several practical ways for an attacker to bypass each of these security mechanisms and successfully replace or alter the iVotronic firmware, without knowledge of any passwords or secret election parameters, possession of a PEB, or breaking any seals. These attacks can be carried out even when the polls are open. It is possible, for example, for a voter (with no inside assistance) to load new firmware into an iVotronic after he or she is finished voting.

We found at least three different vectors that an attacker could exploit to load unauthorized iVotronic firmware under various circumstances.

Via direct replacement of the internal flash chip: The iVotronic terminal housing can be disassembled easily without breaking the seal that protects the CF slot. Disassembling requires only the use of a readily available Torx security screwdriver. Once the housing has been removed, the internal flash chip can be removed from its socket, reprogrammed with a standard flash writer, and replaced. Note that while surreptitious terminal disassembly is unlikely to be possible in an active polling place, it may be an attractive option for an attacker who enjoys unsupervised access to stored terminals (e.g., the night before an election).

Via the firmware update menu: This is the most direct attack against firmware. As discussed before, an attacker can emulate a QA PEB and bypass the password check. If the polls are open, they can be closed by using an emulated QA PEB to clear the terminal first. Note that with this approach, the firmware must be loaded though the external CF card interface, which might be protected with a tamper-evident seal (although that seal can be bypassed by removing the housing).

Via the PEB interface, during the polling day: This is perhaps the most serious practical threat to the iVotronic firmware. As discussed in Section 4.2, errors in the iVotronic's PEB input processing code allow anyone with access to the PEB slot on the face of the terminal (including a voter) to load malicious software that takes complete control over the iVotronic's processor. Once loaded, this software can

[10] Further details about this vulnerability can be found in Sections 7.2.5, 7.2.6 and 7.2.7 of [18].

alter the terminal firmware, change recorded votes, mis-record future votes, and so on throughout the election day and in future elections.

Any compromise of the iVotronic firmware is extremely serious; it can be very difficult to detect whether such firmware has been used in a live election or meaningfully recover once it has. The firmware controls every aspect of the ballot presented to voters, the recorded votes, and the interface to the tally system. Because the RTAL printer is under the control of the firmware, compromised firmware can easily print misleading choices that evade the notice of voters or that cancels the printed ballot (replacing it with other choices) after the voter has left. The discovery of compromised firmware at a terminal casts doubt upon every vote cast at that machine (and, because of additional bugs in the Unity back-end, on the integrity of the results reported county-wide as well).

Compounding the problem is the fact that there are apparently no tools available to counties in the ES&S system that reliably extract or audit the actual firmware present in any given terminal. The iVotronic firmware code includes a number of internal consistency checks intended to detect corrupted firmware. While these checks may be able to detect accidental memory errors, they are ineffective against maliciously installed firmware, which can simply bypass or omit the integrity check functions.

4.3.2 Unity software

No single component of the ES&S system is more important to the integrity of election results than the central Unity election management system. Unity is a complex software suite, consisting of many components that share a common database. Securing a county's Unity system therefore depends on each of its components (and on the computing platforms on which it runs, Windows).

Because Unity (at least as used in Ohio) apparently runs only in a single, secure location in each county, with presumably only trusted staff permitted access to the computers, attack vectors involving unauthorized direct physical access by poll workers, voters or others are a less significant threat here than in precinct equipment sent to the field. However, because the Unity system processes electronic data received from precincts, it is subject to a number of indirect – yet devastating – attacks that can originate with poll workers or voters, even if they cannot themselves physically touch the Unity computers.

Attacks via input from precincts As described in Section 4.2.1, malicious input carried on iVotronic and M100 results media can take over the Unity system when it is loaded for counting. This enables many of the most serious and comprehensive attacks we discovered.

4.3.3 M100 firmware

Firmware can be loaded into the M100 via a specially structured PCMCIA card, the same card used during polling for ballot definitions and other precinct parameters. If new firmware is present on the card when the M100 is turned on, there is a brief screen prompt and the new firmware is loaded. No password is required. The M100 does not perform cryptographic integrity checks on firmware uploads. Any correctly formatted PCMCIA card with M100 firmware (including malicious code created by an attacker) can be installed and accepted as valid. Any poll worker (or other person) with access to the PCMCIA card slot can thus easily load new firmware[11].

Because the firmware is loaded from the same PCMCIA cards used to load ballot definitions, corrupt firmware can also be loaded into the M100 by a corrupted Unity system when an election is provisioned.

M100 firmware controls how ballot definitions are interpreted, the counting and recording of votes, the format of data returned to Unity, and the acceptance and rejection of ballots. The consequences of corrupt M100 firmware are serious, especially given the vulnerabilities in Unity results processing. However, since the paper ballots remain available, they can be recounted if an attack might have occurred.

Unfortunately, as with the iVotronic, there is no mechanism for reliably determining or auditing the actual firmware installed in an M100, so attacks on these devices may be difficult to detect or confirm.

4.4 Ineffective Cryptography and Data Authentication

Much of the critical election data in the ES&S system – ballot definitions, precinct vote tallies, and so on – are communicated between the central county headquarters and precincts through small removable storage media. In iVotronic DRE-based systems, the primary media are PEBs and, in some cases, CF memory cards. In M100-based precinct counted optical scan systems, the primary media are PCMCIA memory cards.

These media share two important characteristics that make them attractive targets for attack: they have no intrinsic security properties of their own and they may pass through many hands on the way to polling places, during the polling day, and back from polling places. That is, it is

[11] Further details about this vulnerability can be found in Section 7.3.1 of [18].

simple to read or alter data on these media, and many people may have the opportunity to do so during an election. For example, iVotronic PEBs are handled by poll workers all through an election day, with memory that can be read or written with a standard palmtop computer and a small magnet. PCMCIA and CF cards, similarly, can be readily read or altered with standard laptop computers.

Data stored on such media should be secured by the use of cryptographic techniques that prevent meaningful access to data without knowledge of the correct key.

Unfortunately, the ES&S system does not employ cryptography at all in the M100-based optical scan system. The iVotronic DRE system does use cryptography, but errors in its implementation render the protection completely ineffective. The lack of effective cryptographic protection enables a large fraction of the exploitable vulnerabilities in the whole system.

Unauthenticated M100 data M100 PCMCIA cards are used to load ballot definitions and firmware into the M100 and to report tallies back to the Unity system. The data for each of these functions are not cryptographically protected; an attacker with access to an M100 PCMCIA card can easily forge or modify this data. A linear cyclic redundancy code (CRC) is included with the PCMCIA data, but an attacker can easily calculate this; CRC codes are not keyed and are not designed to provide security against deliberate data modification[12].

Ineffective iVotronic cryptography The iVotronic DRE uses the Blowfish [22] cipher to protect data stored on the PEB and the CF card. Unfortunately, the manner in which the encrypted data are stored on the PEBs effectively neutralizes the cryptographic protection. The PEB contains an EQC, encoded using an unkeyed (non-cryptographic) algorithm. The EQC is used to encrypt the Blowfish key, which is used to the encrypt the rest of the data on the PEB. Remarkably, both the EQC and the encrypted key (which, again, is encrypted using the EQC) are present on the PEB. Although much of data on the PEB is encrypted, there is sufficient unencrypted information stored along with it that allows an attacker to trivially discover the key[13].

Lack of results integrity in Unity The obvious attacks enabled by the lack of cryptographic protection of precinct media include alteration or forgery of data, unauthorized loading of firmware, as discussed in the rest of this report.

Additional vulnerabilities are introduced by Unity's poor validation of various reported precinct data. In particular, the precinct results reported on an incoming M100 PCMCIA card are not checked against the precincts for which the card was originally provisioned. This allows anyone with access to a card to add tally results for extra precincts, which will be added to (or supplant, depending on the mode the Unity operator is using) the true precinct results when read into the database, as described in Section 4.2.1.

5 Viral Propagation through Component Interaction

The software or firmware of almost every major component of the ES&S system can be altered or replaced by input from the other components with which it communicates. In particular, note that, by design or software flaw:

- The Unity system software can be modified by election results media originating from iVotronics and M100s (due to Unity buffer overflows)
- The iVotronic firmware can be modified by configuration media originating from the Unity system (due to iVotronic buffer overflows).
- The M100 firmware can be modified by configuration media originating from the Unity system (due to the design of the M100 firmware management functions).
- A compromised iVotronic can modify a PEB such that it carries a malicious payload which infects other iVotronics on which it is subsequently used. This iVotronic to iVotronic propagation can happen, for example, while a master PEB is being used to run *Logic-and-Accuracy* tests on the iVotronic terminals being used in a particular election.

This confluence of vulnerabilities creates a "closed loop" for viral propagation into every part of the ES&S system through the compromise of a single system component. The viral loops in the system are depicted on Figure 5.

For example, a voter can compromise an iVotronic terminal though its PEB slot. The iVotronic, then, may be programmed to create results media (at the end of the election day) that, in turn, corrupts the software of the central Unity system. The compromised Unity system, in turn, may be programmed to load corrupted firmware into all

[12] Further details about this vulnerability can be found in Sections 7.3.3 and 7.3.6 of [18].

[13] Further details about this vulnerability can be found in Section 7.2.1 of [18].

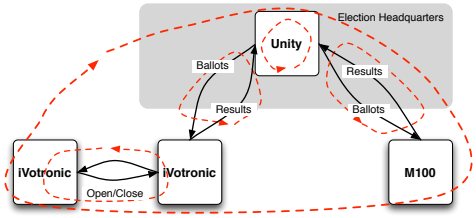

Figure 5: Viral loops through ES&S components

M100s and iVotronics in the county when provisioning a subsequent election. At this point, every major component of the system is running compromised code, which originated with a single attacker with only voter access in a single precinct. Needless to say, such an attack represents a grave threat to the integrity of the elections of any jurisdiction to which this happens[14].

6 Reviews by Commercial Consultants

In addition to our analysis, two commercial contractors examined the ES&S systems as part of Project EVEREST. SysTest Labs, a NIST approved testing lab performed audits of the documentation, county election procedures and the hardware, software and firmware versions. A professional security assessment company, MicroSolved, Inc (MSI) performed independent vulnerability analysis and penetration testing.

The MSI team applied a systematic evaluation methodology of attack surface mapping, threat modeling, and poor trust/cascading failure analysis to assess where to focus their attention, and then used standard pen-testing tools including attacking physical security, network scanning and 'fuzzing' to locate and exploit several vulnerabilities in the ES&S system. These methods were especially successful at attacking the Windows 2003 servers and Windows XP Professional workstations used to manage the election and host the Unity software, and at finding the physical vulnerabilities that would allow an attacker access to the system internals.

These methods were less successful in exploiting the vulnerabilities found in the ES&S components. For example MSI noticed that a simple magnet could activate the iVotronic DRE, and the MSI team was also able to access many of the open, unprotected ports, but were unable to tamper with the serial protocols. Because of this, MSI was not able to demonstrate that many of the attacks they envisioned in their threat model were actually practical rather than merely possible. Therefore they were unaware of the vulnerability of Unity to an attack on any of it's individual components, and they couldn't discover the more serious vulnerabilities which involve interactions between components and could lead to viral propagation.

[14]This attack scenario is described in detail in Section 9.3.11 of [18].

7 Conclusion

As detailed in the previous sections, we found significant and pervasive vulnerabilities throughout the ES&S system. In both optical scan and DRE configurations the system suffers from vulnerabilities that can be relatively easily exploited by individual poll workers or voters (at precincts and under election conditions) that not only compromise the results from individual machines, but that can inject arbitrary malicious code into the back end tally system that reports the official county-wide election results. Several closed viral loops are present, allowing, under some conditions, a single compromise of a single precinct machine to permanently control the entire county election system. Audit mechanisms that might detect and recover from such attacks are easily defeated or not present at all. For example, there is no mechanism for extracting and auditing the firmware installed in an iVotronic or M100.

By themselves, the weaknesses reported here represent serious practical concerns; we found, after all, exploitable vulnerabilities in a widely fielded e-voting system used across the US and elsewhere. But perhaps an even more serious concern is the systemic failure – at every stage – of the various standards, certification and testing processes that were intended to prevent these vulnerabilities from appearing in the first place.

We note in particular that, in contrast to the "official" Independent Testing Authority (ITA) practices, we followed no particular methodology or standard practices in conducting our experiments and analysis. Because of the very limited time and other resources available to us, we adopted an almost entirely *ad hoc* approach, focusing our attention on those parts of the system that we believed might harbor exploitable vulnerabilities. While we used some source code analysis tools (e.g., *Fortify),* we applied them only selectively. That is, rather than a "certificational" process in which the evaluator checks for compliance with a finite set of criteria, we adopted the triage strategy of an attacker, seeking out weaknesses in the places we thought we would be most likely to find them and moving on to the next.

Our approach has the disadvantage of depending, to a much larger extent than the certificational approach, on the expertise (and luck) of the analysts. And a negative results from such an analysis, where no vulnerabilities are found, would say little about whether any are present. But

in our case the approach paid off handsomely.

It is worth noting that, in parallel to our study, the State of Ohio contracted two private consultancies to perform more traditional certification studies of the ES&S system. While the scope of and resources provided to these studies (by SysTest and MicroSolved) were not completely identical to ours, the overall goal was the same: a security assessment of the system identifying specific deficiencies. While commercial certificational studies reported a few problems that we missed, the vast majority of the most sweeping and systemic vulnerabilities were found only through our *ad hoc* analysis.

Acknowledgments

This paper draws primarily from our work on Project EVEREST [18], in particular Chapters 4, 5, 6, 7 and 8. Penetration testing and attack scenario development for ES&S was preformed by the WebWise Security team: Giovanni Vigna, Richard Kemmerer, Davide Balzarotti, Greg Banks, Marco Cova, Viktoria Felmetsger, William Robertson, and Fredrik Valeur. We held numerous discussions and collaborations with the Penn State team: Patrick McDaniel, Kevin Butler, William Enck, Harri Hursti, Steve McLaughlin, and Patrick Traynor, and with our policy consultants Joseph Lorenzo Hall and Laura Quilter. Infrastructure for the U. Penn voting laboratory was supported in part by a gift from Barbara and Morris Pearl. Project EVEREST was sponsored by Ohio Secretary of State Jennifer Brunner.

References

[1] ES&S About. http://www.essvote.com/HTML/about/about.html.

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

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

[4] About ES&S: Did You Know? http://www.essvote.com/HTML/about/dyk.html.

[5] Atsec Information Security Corporation. ES&S Unity 3.0.1.1 source code review, February 2008. http://www.sos.ca.gov/elections/voting_systems/unity_3011_source_code.pdf.

[6] Matt Blaze, Arel Cordero, Sophie Engle, Chris Karlof, Naveen Sastry, Micah Sherr, Till Stegers, and Ka-Ping Yee. Source code review of the sequoia voting system. University of California, Berkeley under contract to the California Secretary of State, July 2007.

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

[8] California Secretary of State. Top-To-Bottom Review, July 2007. http://www.sos.ca.gov/elections/elections_vsr.htm.

[9] Ariel Feldman, J. Alex Halderman, and Edward Felten. Security analysis of the Diebold AccuVote-TS voting machine. In *Proceedings of the 2007 Usenix/ACCURATE Electronic Voting Technology Workshop*, August 2007.

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

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

[12] Harri Hursti. The Black Box Report: Critical Security Issues with Diebold Optical Scan Design. Black Box Voting, July 2005.

[13] Harri Hursti. Diebold TSx Evaluation: Critical Security Issues with Diebold TSx. Black Box Voting, Unredacted release July 2, 2006, May 2006.

[14] InfoSENTRY Services, Inc. Volume 1, Computerized Voting Systems, Security Assessment: Summary of Findings and Recommendations . For Ohio Secretary of State, November 2003.

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

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

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

[18] Patrick McDaniel, Matt Blaze, and Giovanni Vigna *et al. EVEREST: Evaluation and Validation of Election-Related Equipment, Standards, and Testing (academic report)*. For the Ohio Secretary of State, December 2007.

[19] Microsolved, Inc. *EVEREST Project: ES&S System Executive Summary Report*. For the Ohio Secretary of State, 2007.

[20] RABA Innovative Solution Cell (RiSC). Trusted Agent Report: Diebold AccuVote-TS Voting System. RABA Technologies LLC for the Maryland General Assembly, January 2004.

[21] Dan Rather. The trouble with touch screens. HDNet: Dan Rather Reports, August 2007. http://www.hd.net/drr227.html.

[22] Bruce Schneier. Description of a new variable-length key, 64-bit block cipher (Blowfish). *Fast Software Encryption: Cambridge Security Workshop Proceedings (December 1993)*, LNCS 809:191–204, 1994.

[23] Giovanni Vigna, Richard Kemmerer, Davide Balzarotti, Greg Banks, Marco Cova, Viktoria Felmetsger, William Robertson, and Fredrik Valeur. Security Evaluation of the Sequoia Voting System: Public Report; UC Red Team Report: Sequoia Voting Systems. University of California, Berkeley under contract to the California Secretary of State, July 2007.

[24] David Wagner, David Jefferson, and Matt Bishop. Security Analysis of the Diebold AccuBasic Interpreter. Voting Systems Technology Assessment Advisory Board (VSTAAB), February 2006.

[25] Alec Yasinsac, David Wagner, Matt Bishop, Ted Baker, Breno de Medeiros, Gary Tyson, Michael Shamos, and Mike Burmester. Software review and security analysis of the ES&S iVotronic 8.0.1.2 voting machine firmware. For the Florida Department of State, February 2007.

Appendix: Software Versions

The source code analysis and red teams were provided with the following versions of the Unity environment and source code by the vendor:

Component	**Application Version**
Unity	3.0.1.1
Audit Manager	7.3.0.0
Election Definition Manager	7.4.4.0
Election Reporting Manager	7.1.2.1
Hardware Program Manager	5.2.4.0
Data Acquisition Manager	6.0.0.0
ESS Image Manager	7.4.2.0
iVotronic Image Manager	2.0.1.0
iVotronic Firmware	9.1.6.2 9.1.6.4
M100 Firmware	5.2.1.0
M650 Firmware	2.1.0.0
RMCOBOL RT	7.5.01
COBOL WOW RT	3.12