

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 III

Analysis of the Premier Elections Solutions, Inc. Voting Systems

PREMIER EXECUTIVE SUMMARY

The study included in this part of the EVEREST report evaluates the ability of the Premier voting system to guarantee a trustworthy election. The review team was provided access to the Premier 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 analysis suggests that the Premier system lacks the technical protections necessary to guarantee a trustworthy election under operational conditions. Flaws in the system's design, development, and processes lead to a broad spectrum of issues that undermine the voting system's security and reliability. The resulting vulnerabilities are exploitable by an attacker, often easily so, under election conditions. These vulnerabilities are the result of the following failures of the Premier system's design or implementation:

- *Failure to effectively protect vote integrity and privacy* - Numerous vulnerabilities allow an attacker to modify or replace ballot definitions, to change, miscount, or discard completed votes, or to corrupt the tally processes. Further issues expose voter choices and can lead to voter coercion and vote selling.
- *Failure to protect election from malicious insiders* - The Premier system does not provide adequate protections to ensure election officials, poll workers, or vendor representatives do not manipulate the system or its data. These attacks are often invisible after the fact, and therefore misuse is difficult or impossible to uncover later.
- *Failure to validate and protect software* - The Premier system makes only limited and often ineffective attempts to validate the software running within system. Thus, an attacker may exploit software and replace it with their own with little fear of detection. Further, the recommended means of installing and upgrading software is frequently highly dangerous.
- *Failure to provide trustworthy auditing* - The auditing capabilities of the Premier 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.
- *Failure to follow standard software and security engineering practices* - A root cause of the security and reliability issues present in the system is the visible lack of sound software and security engineering practices. Examples of poor or unsafe coding practices, unclear or undefined security goals, technology misuse, and poor maintenance are pervasive. This general lack of quality leads to a buggy, unstable, and exploitable system.

We found the Premier software to be unstable. Frequent crashes, system lock-ups, and unexplained errors were commonplace in our experiments. Stability problems were acute in the GEMS server, where failures occurred during normal use and under limited loads.

Our findings are consistent with those of previous studies. When taken as a whole, this and previous studies highlight a central point of concern: there is a demonstrative lack of improvement in the security of elections conducted using the Premier system. Initial reviews of the Premier system were undertaken as early as 2001. After six years of reviews and many new software and hardware upgrades, reviewers not only continue to find the same and similar problems as reported earlier, but continue to uncover new serious issues. Thus, the only reasonable conclusion that one can draw is the engineering approaches undertaken by Premier to eliminate previous problems and avoid new ones are failing.

The flaws in the Premier system place the security of an election almost entirely on physical procedures. Our analysis suggests that when those practices are not uniformly followed, it will be difficult to know when attacks occur. Even when the attacks are identified, it is unlikely that the resulting damage can be easily contained and the public's belief in the accuracy and fairness of the election restored.

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

PREMIER STUDY OVERVIEW

11.1 Part Structure

The following chapters detail the Premier 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 Premier system only*. None of the included material should be considered to apply to either the ES&S or Hart systems, nor should any statement be construed to be making any quantitative or qualitative comparisons between 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 Premier system and its use in Ohio elections. Chapter 12 broadly characterizes the security and reliability of the Premier system by highlighting several architectural and systemic issues encountered in the review. Chapter 13 provides detailed technical information on the confirmation of previously reported issues and Chapter 14 provides similar detail on new issues uncovered by our analysis. Chapter 15 describes a number of attack scenarios that demonstrate how the identified issues may be used in concert to subvert or otherwise indict an election.

11.2 Architecture

The purpose of this section is to familiarize the reader with both the architecture of the Diebold/Premier Electronic Voting Systems. The contents of this report have been taken from a variety of public documents and personal experience, and are not intended to be an exhaustive or authoritative description of the system.

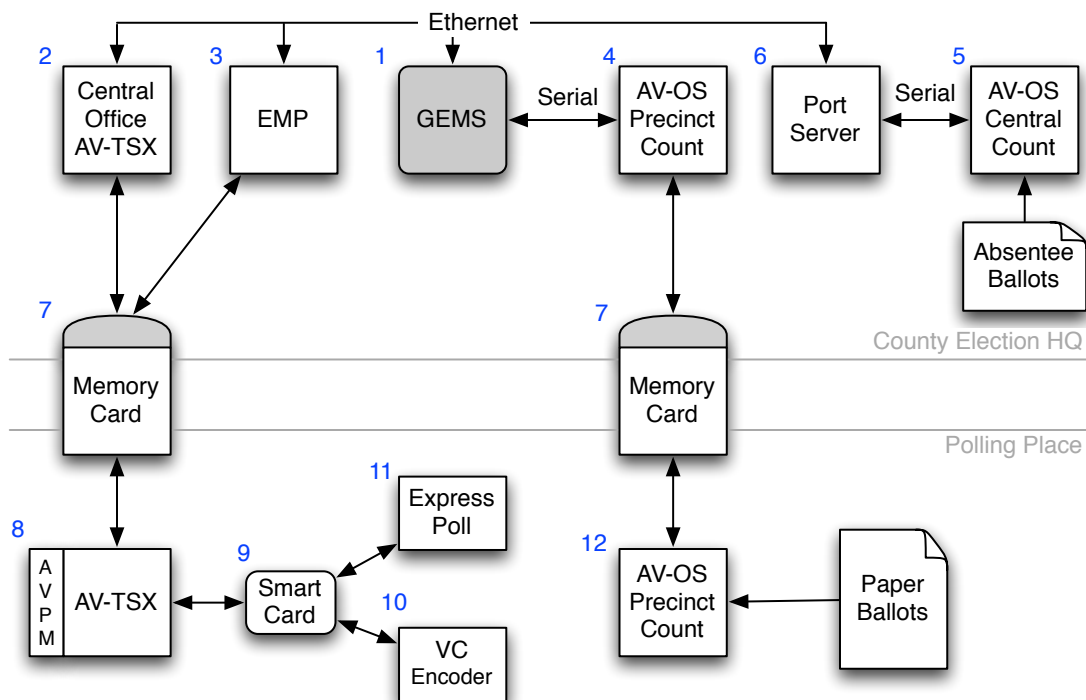


Figure 11.1: The major components and their interconnections for counties using Premier Electronic Voting Systems. Unless otherwise indicated, arrows depict physical transport of cards and ballots.

11.2.1 Components at County Election Headquarters

A number of voting components reside at the county election headquarters. Refer to Figure 11.1 for a pictorial representation of component interaction.

1. **Global Election Management System (GEMS):** The GEMS server is responsible for running back-end election processes. Accordingly, it is stored at the county's election headquarters. Election administrators use GEMS to create ballot definitions, program memory cards and tally all votes when an election closes. It is connected to an AV-OS Precinct Count via serial cable and either the Election Media Processor (EMP) or the Central Office AV-TSX in the county election headquarters by either a dedicated or office-wide LAN.

The GEMS server runs on a server-class machine running Windows Server 2000. The Premier software typically running on this physical server includes:

- **GEMS:** Software used to define, tally, and report an election.
- **KeyCardTool:** Software used to create administrative smart cards, including the *Security Key Card*, the *Central Administrator Card*, and the *Supervisor Card*.

To harden the GEMS server against previously reported vulnerabilities, Ohio GEMS servers additionally include third party security software: *Verdasys Digital Guardian*, *Sygate Security Agent* network firewall, and *McAfee VirusScan*.

2. **Central Office AccuVote Touchscreen (AV-TSX):** The Central Office AV-TSX is a touchscreen voting device used to write ballot definitions and software updates to memory cards before an election and read election results from memory cards after an election. Any AV-TSX can be used to support these tasks. However, counties deploying many memory cards may optionally use the EMP.

The Central Office AV-TSX runs on the AV-TSX hardware with the addition of an Ethernet card that plugs into a memory card slot. The central office mode of operation is selected by an administrator. The device runs Microsoft Windows CE and includes the following Premier software:

- **BallotStation:** Software through which election officials create memory cards (same software as voters make their choices).
 - **Bootloader:** Software used to load the operating system on the AV-TSX hardware.
3. **Election Media Processor (EMP):** Because efficiently encoding memory cards before and reading election results after an election is difficult using Central Office AV-TSXs (which can only read a single PCMCIA card at a time), Premier offers the EMP server. This device can read as many as six memory cards in parallel, significantly speeding up the above operations.

The EMP server can run on either a Windows 2000 or Windows XP machine. Premier software running on this physical server includes:

- **EMP:** Software used to communicate with GEMS and interface with the memory cards.
4. **AccuVote Optical Scan (AV-OS) Precinct Count:** The AV-OS Precinct Count (AV-OS PC for short) machines use an older style memory card incompatible with those used by the AV-TSX and EMP. Therefore, the central office contains a dedicated AV-OS PC in order to write ballot definitions to memory cards before an election and read election results after an election. The central office AV-OS PC software and hardware are identical to the units used at the polling place.

The AV-OS PC does not have a dedicated operating system. The Premier software running on this physical machine includes:

- **AV-OS PC:** Software used to operate the AV-OS hardware, read ballots, and tally results.
5. **AccuVote Optical Scan (AV-OS) Central Count:** The AV-OS Central Count (AV-OS CC for short) is an optical voting scanner used read and record absentee ballots. The AV-OS CC uses identical hardware to the AV-OS PC; however, it runs different software. The only difference between the AV-OS PC and AV-OS CC is the installed memory chips. The AV-OS CC communicates with the GEMS server throughout its operation using the Port Server, which changes the serial connection to an Ethernet connection.

The AV-OS CC does not have a dedicated operating system. The Premier software running on this physical machine includes:

- **AV-OS CC:** Software used to operate the AV-OS hardware, read ballots, and communicate with the GEMS server.
6. **Port Server:** The Port Server connects machines using serial port communication to Ethernet networks. Premier uses the Digi PortServer II at the county election headquarters, which allows them to connect up to 16 AV-OS CC machines to the GEMS server. AV-TSX units located at the county election headquarters use an Ethernet card plugged into a memory card slot and therefore do not need to use the Port Server to communicate with GEMS.

7. **Memory Cards:** Premier Election Systems rely on memory cards as the major avenue of communication between the back-end servers and the polling place. Before an election, for instance, an AV-TSX will read the card to find the ballot definition, sound files, translations into other languages (.edb and .xtr files), interpreted code (.abo files) used to print reports and other configuration information. Votes are also stored on memory cards (.brs files), which are returned to the county election headquarters for tabulation at the close of an election.

Premier systems rely on two types of memory cards. As described above, all AV-TSX units use standard 128 MB PCMCIA memory cards. Such cards are readily purchased at any electronics retailer, and all files can be read by any PC with a PCMCIA card reader, commonly found on laptops. The AV-OS units rely on a 128 KB EPSON 40-pin memory card. Manufacturing of this card ceased in 1998. Because two different technologies are used, the AV-TSX memory cards cannot be used in an AV-OS machine and visa-versa.

A Premier election system may also include other components at the county election headquarters. However, for various reasons, the following components are not used in Ohio. We list them for completeness.

- **AccuFeed:** The AccuFeed unit is used in the county election headquarters to assist with the tallying of paper ballots. Specifically, this device helps feed a large number of paper ballots to the AV-OS Central Count unit. This unit is not used in Ohio and is therefore not examined in this study.
- **Modem Connections:** Communication between voting machines deployed in precincts and the servers in the county's election headquarters is possible using modems. While this practice is strictly forbidden by law in Ohio, all voting machines contain the necessary software and fully enabled hardware to perform such operations. In some precincts, use of these devices is inhibited using a plastic or screw-attached metal cover.

11.2.2 Components at Polling Place

The remaining voting components reside at the polling place. Refer to Figure 11.1 for a pictorial representation of component interaction.

8. **AccuVote Touchscreen (AV-TSX):** As previously mentioned, the Precinct Count AV-TSX is exactly the same as the Central Office unit (i.e., any unit can be used for either mode). These units are deployed in each polling place and allow for touchscreen voting to occur. All AV-TSX machines used in Ohio also include an AccuVote Printer Module (AVPM), also known as a Voter Verifiable Paper Audit Trail (VVPAT) printer unit, which creates a physical copy of a cast ballot on thermal paper.

The AV-TSX runs Microsoft Windows CE and includes the following Premier software:

- **BallotStation:** Software through which voters make their choices.
 - **Bootloader:** Software used to load the operating system on the AV-TSX hardware.
9. **Smart Card:** The AV-TSX determines what a user can do based on the inserted smart card. A smart card is a credit card like device that contains a small chip capable of performing cryptographic operations. The smart cards come in four varieties: *Voter Access Card*, *Supervisor card*, *Central Administrator card*, and *Security Key Cards*. When one of these cards is inserted into an AV-TSX, the person using this card is allowed to access an associated set of operations on the machine. For instance, a user entering a Voter Access Card would not be allowed to perform administrative operations.

10. **Voter Card Encoder (VCE):** The VCE is a standalone, calculator-like device used to make *Voter Access Card* smart cards valid for casting one ballot. The VCE is used by polling places that use traditional paper voter log books to identify legitimate registered voters.

The VCE includes the following Premier software:

- **VCEncoder:** Software used to read and write Voter Access Cards.
11. **ExpressPoll:** The ExpressPoll tablet devices functions as an electronic replacement for the traditional voter log book. As users enter the polling place, a poll worker can confirm their eligibility and issue voter smart cards.

The ExpressPoll runs Windows CE and includes the following Premier software:

- **ExpressPoll:** Software that allows a poll worker to find registered voters in a electronic database.
 - **CardWriter:** Software that allows the Express Poll to write to inserted smart cards.
12. **AccuVote Optical Scan (AV-OS) Precinct Count:** The AV-OS Precinct Count (AV-OS PC for short) voting machine is primarily used in the polling place (the AV-OS PC described above exists solely to create memory cards to be used at the polling place). Voters filling out paper ballots in their precinct use this device to scan and tabulate their results. After scanning a ballot, The paper copy of the ballot is dropped into a locked plastic bin in case an audit is required.

The AV-OS PC does not have a dedicated operating system. The Premier software running on this physical machine includes:

- **AV-OS PC:** Software used to operate the AV-OS hardware, read ballots, and tally results.

11.3 Election Procedures

Given the election headquarters and polling place components described in Sections 11.2.1 and 11.2.2, we now discuss how an election proceeds. Note that these procedures vary from county to county.

11.3.1 Pre-Election Procedures

- Before an election, the election administrators define the ballot. These definitions are created on the GEMS server.
- When ready, the GEMS server contacts the AV-OS PC, AV-TSX, and optionally EMP machines residing in the central office via the appropriate serial cable and network interfaces. These devices then receive the ballot definitions.
- The central office AV-OS, AV-TSX, and EMP machines encode the ballots onto memory cards. One memory card must be encoded per machine in the county. Memory cards encoded for touchscreen voting machines are not physically compatible with those encoded for optical scanning machines.
- Memory cards are then sent to each polling site. The details of how they get there, however, vary. Administrators can insert the cards into the appropriate machines before they are delivered to the polling place. Alternatively, cards can be delivered separately and inserted by poll workers.

11.3.2 Polling Place Election Procedures

The polling place election procedures for the AV-TSX proceed as follows.

- An election begins when a polling place administrator inserts a supervisor smart card. This card allows the administrator to open and close elections and also view machine logs.
- When voters arrive at the polling place, poll workers ensure they are legitimate registered voters using either the traditional paper voter log or the ExpressPoll.
- After voters are authenticated, they receive a *Voter Access Card* smart card, which allows them to cast a single ballot. With this card in hand, the voter approaches the AV-TSX.
- Upon reaching the AV-TSX, the voter inserts the *Voter Access Card* into the machine and follows the on screen instructions to cast a ballot. Before the ballot is officially cast, the voter is asked to confirm the available print-out is correct.
- After the ballot has been recorded by the machine (and stored on the memory card), the AV-TSX reprograms the smart card so that it can not be used until re-encoded by either the VCE or ExpressPoll.
- At the close of election, the polling place administrator closes the election by again inserting the supervisor smart card and selecting the appropriate option.

The polling place election procedures for the AV-OS Precinct Count proceed as follows.

- AV-OS machines are configured by inserting one of the above mentioned memory cards. Machines then run the Zero Report to demonstrate that they do not currently contain votes.
- Voters fill out ovals on the paper ballots at the polling place. When finished, ballots are inserted into the AV-OS PC. A paper feeder similar to a fax machine is used to take the ballot for optical scanning. Ballots are then physically stored within the AV-OS until the election concludes.
- Users are notified if their ballots are incorrectly filled out. For instance, if a voter selects two candidates in a race where only one candidate can be chosen, the ballot will be returned and the voter will be instructed to have the ballot invalidated/voided. The voter can then receive a new paper ballot.
- At the close of election, the polling place administrator unlocks the top panel of the ballot box and holds down the two buttons while feeding the scanner a special ender card. This action closes the election and causes the AV-OS PC to print an election summary.

11.3.3 Post-Election Procedures

- Votes can be sent to the County Election Headquarters in one of two ways. One option is to remove the memory cards and physically transport them to the central office. Alternatively, the entire voting machine can be transported. Note that votes can also be reported via modem; however, Ohio law prohibits the use of modems with election equipment.

- Once memory cards arrive at the central office, the AV-TSX memory cards are inserted into either the central office AV-TSX or the EMP, and the AV-OS PC memory cards are inserted into the central office AV-OS PC. Note that the memory cards are not physically compatible, so an AV-OS machine can not be used to tally votes from an AV-TSX. The memory card contents are sent to the GEMS server for tabulation.
- Once all memory cards are reported to the GEMS server, an official election results summary is printed.

11.4 Verdasys Digital Guardian

Due to the results of previous studies of the Diebold/Premier elections equipment, the state of Ohio has requested that Premier include additional third party security software to harden the GEMS server. Specifically, the GEMS server setup in Ohio includes: *Verdasys Digital Guardian*, *Sygate Security Agent* network firewall, and *McAfee VirusScan*. The latter two security tools provide standard system protection and warrant little discussion. However, Digital Guardian is presented as a remedy to a number of significant GEMS vulnerabilities such as the ability for an attacker to perform arbitrary modification of an election database simply by having access to the GEMS server filesystem (see Issue 13.1.2). We were unable to find significant technical specifications of Digital Guardian from public resources. Therefore, we performed penetration testing of the Digital Guardian protected GEMS server. Note that because Digital Guardian is considered COTS software, Premier was not required to provide any source code, nor were we provided any technical documentation describing how the system works. However, we were provided the current policy specifications and some notes from a Premier technician, which greatly aided our understanding of how Digital Guardian protects a system.

Digital Guardian was designed to protect a system running Windows 2000 or XP. It allows an administrator external from the local system to specify policies that control how all local users are allowed to execute programs and access files. In Ohio's setup, a state employee possesses a special laptop called the Digital Guardian console. Each GEMS server contains the Digital Guardian Agent enforces the policy specified by the console. The only way the Digital Guardian Agent can be disabled is if a state employee directly connects the Digital Guardian console the GEMS server and specifies that the agent should be disabled.

The Digital Guardian Agent running on all GEMS servers enforces two high level policies (keep in mind that in Ohio, all county GEMS servers are administered by Premier employees). First, the election databases should only be accessed by the GEMS program. Second, the vendor employee should not have access to any GEMS data. The remainder of the policy installed in each Digital Guardian Agent exists purely to retain the system integrity and keep an attacker from circumventing Digital Guardian.

In order to provide separation between users, three Windows users have been created: *Administrator*, *GEMSAAdmin*, and *GEMSUser*. The *Administrator* account performs basic administration and maintenance of the GEMS server, but operations that involve GEMS data are forbidden. The *GEMSAAdmin* account is not a system administrator, rather, it is the only user allowed to perform file manipulation operations, e.g., copy, move, delete, on the election database files. Finally, the *GEMSUser* account may only modify election database files using the GEMS program, and it should not be able to delete, copy, or paste the files. Additionally, *GEMSUser* is allowed to burn backups of the election database, as this is a necessity on election day.

Our study investigates Digital Guardian's ability to enforce the high level protection policies. In doing so,

we concentrated on discovering ways to disable Digital Guardian. In addition, we considered ways in which users could perform actions explicitly denied by the policy. Finally, our analysis only considers the GEMS server. While we were provided a Digital Guardian console laptop, we were unable to analyze network communication due to lack of time. We recommend future investigations thoroughly study ways in which an attacker can remotely compromise or bypass Digital Guardian.

PREMIER SYSTEMIC AND ARCHITECTURAL ISSUES

This report examines the Premier system's resistance to manipulation of electoral outcomes, procedures and public perception. We have found substantial flaws that seriously inhibit the ability of the Premier system to meet these goals. Many of the issues presented in the following chapters relate to critical functional or procedural problems embodied in the Premier design and implementation. Such problems undermine the security of the system and provide practical avenues to compromise critical election data and systems. This chapter broadly characterizes¹ the report's findings by highlighting the five areas of most concern/interest:

- **Ineffective Data and Privacy Security** - The methods used to protect the integrity and privacy of important election data are circumventable, often trivially.
- **Ineffective Cryptographic and Hardware Security** - The use of many standard security technologies is deeply flawed.
- **Unsafe Software Management** - The means by which the software of election equipment is installed and upgraded are often highly dangerous.
- **Design and Code Quality Problems** - Universal poor design and coding practices lead to a brittle system that is a potentially irreversible source of security problems.
- **Previously Unreviewed Components** - This chapter concludes with a brief characterization of four previously unreviewed Premier components: the EMP server, Digital Guardian, ExpressPoll and the Voter Card Encoder.

The findings detailed in this and the following chapters are fully consistent with previous studies. In particular, we found the architectural and systemic findings of the Premier Source Code Report of the California TTBR study to be universally true. Here, we build upon those observations and attempt to more generally characterize the specific classes of security issues within the Premier 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 issues that can have serious negative consequences for real elections.

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 13.2.1 discusses the lack of protections on Premier memory cards, and is referenced as "(13.2.1)". 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.

12.1 Ineffective Data and Privacy Security

12.1.1 Memory Cards

Memory cards are the central device for storing and communicating election data. The cards are programmed by GEMS, carried to the polling place, used to collect voter choices during the election, carried back to the election headquarters, and tallied at the end of the election. We found the memory card protections were ineffective at preventing an attacker from viewing and modifying data held on these cards.

Memory cards in the AV-OS are unprotected (13.2.1)—an adversary who gains access to a card may modify it in arbitrary ways, by creating or eliminating votes, changing ballot definitions, or even modifying the programs that run on the election devices (13.2.10, 13.2.11). Attempts to protect the card are ineffective (13.2.3, 13.2.5). Memory cards in the AV-TSX are protected using a *Data Key*. However, this (and other) keys are not adequately protected (13.3.12, 13.3.4)—thus, an attacker can extract the key from Premier devices and perform the same forgery/modification as in the AV-OS.

12.1.2 Voter Privacy

One of the core requirements of an election is voter privacy; a voter's choices should not be exposed to anyone, including election officials. The Premier system failed to meet this goal. On the AV-TSX, the votes are stored in order on the memory card (13.3.19), with a timestamp in the VVPAT (13.3.20), and in memory with a serial number that is generated deterministically using the system's Data Key (13.3.21). Any one of these vote representations can be used in conjunction with poll books and observed voter order to determine voter choices.

The AV-OS does not record individual votes, but tallies them as they are read from the optical scan. The vote order is retained by the stacking of ballots in the ballot box (14.4.2). By exploiting this, an attacker could recover voter choices as in the AS-TSX.

12.1.3 GEMS

The GEMS server is a central repository for election data. It maintains the election data in database files on the local hard disk (with the file extension `.mdb`). These databases are largely unprotected, and can be freely accessed (13.1.2). The review team built an application to access and modify the election database within a few minutes using simple and widely available Visual-Basic tools. Moreover, GEMS uses the Microsoft Jet interface to manipulate databases (13.1.1), whose use in election systems has been advised against by Microsoft for stability and accuracy reasons.

The GEMS implementation also provides other means of exploiting the systems and gaining access to the internal data. For example, GEMS does not provide adequate input filtering from fields returned from the database (13.1.7). This can be used to launch buffer overflow attacks. There are numerous other methods the exploit errors in the system to compromise GEMS and therefore alter internal data (and the databases) with impunity (13.1.4, 13.1.6, 13.1.9, 14.3.1).

Access to the GEMS functionality is governed by passwords that can be extracted and cracked using standard password cracker tools, or simply by adding an attacker "user" into the GEMS database and giving them administrator rights (13.1.8). Such access can be used to manipulate election data and results through normal

Premier interfaces, possibly arousing less suspicion than other more invasive activities.

12.2 Ineffective Cryptographic and Hardware Security

12.2.1 Key Management

Cryptographic keys are used in AV-TSX, EMP, and GEMS to preserve the secrecy and integrity of election data, and for securing communication between devices. We have found that the creation, storage, and use of the keys is insufficient to ensure an attacker cannot view or modify election data.

The AV-TSX uses several keys to implement security; a *data key* encrypts votes on the memory card, a *smart card key* used to authenticate the AV-TSX to a smart card, and a *system key* used to secure the keys. The data and smart card keys are placed in a *key file* encrypted with the system key in the AV-TSX.

The mechanisms used to store these cards on the AV-TSX are insufficient to prevent an attacker from obtaining them. The system key is derived from the serial number printed on the side of the voting device, and is therefore easily attainable (13.3.5). An attacker that gains access to the filesystem on the AV-TSX (by exploiting, for example, a buffer overflow) can retrieve the key file and readily recover the other keys—thus enabling any number of other attacks on the election and results.

Note that the data key is the same for all devices in the county (14.1.2). This is poor security design—compromise of any one of potentially hundreds of devices is sufficient to subvert an entire county. Here the Premier 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.

The EMP and AV-TSX devices can directly communicate with GEMS over a network. This communication can be secured using SSL, a popular and secure communications protocol commonly used for web applications. However, the keys used are the same within the county (14.1.6) and are protected with the weak fixed password “diebold” (13.3.11).

12.2.2 Password Management

GEMS governs access to important system features via user logins. Users logging into the system provide a username and password. Each user has a set of rights to system functions, e.g., backup, election activities, auditing. Users defined as having administrator rights generally have more rights than non-administrator users. The users and their passwords are maintained in a database file on the local operating system. In Ohio, these files are protected from misuse by the Digital Guardian (DG) application.

DG can be disabled or bypassed (14.7.1, 14.7.3, 14.7.4) by a user who has access to the GEMS desktop (i.e., is able to login to Windows). Thereafter they can freely extract password data (hashes) from the database by accessing the files directly (13.1.8). Once retrieved, an attacker can use widely available password crackers to recover the unhashed password. However, this process can take hours or days on a powerful computer. The attacker may decide instead to simply replace a target victim user’s password with one of its choosing. Thereafter, they could login under the victim’s account, perform whatever operations they choose, and logout. If they desire to cover their tracks, they could restore the original password. They can thus become any back-end user and perform arbitrary election management operations *without ever learning any GEMS password*.

The AV-OS supervisor PIN (used to gain access to the administrative menus on the AV-OS) can be read directly from the memory card (it is obfuscated in a trivially breakable way) (13.2.8). The Smart Card PIN (used to authenticate smart cards for the AV-TSX) programmed into each security card is not recoverable by any technical means that we know of.

12.2.3 Hardware Tokens

Voter smart cards are used in part to authenticate users to the AV-TSX. The Voter Card Encoder (VCE) is a handheld device that creates these “voter cards”. The poll worker inserts a smart card into a slot on the device. The VCE then encodes them with authentication information that the AV-TSX recognizes using a cryptographic key. The smart card is given to the voter, who carries it to the AV-TSX to begin voting. The smart card allows the user to vote once.

The encoding process is rather open—once enabled, the VCE remains enabled until it is turned off (14.5.2). Thus, an attacker able to obtain an enabled VCE can manufacture any number of cards. Moreover, if proper procedures are not followed, default poor keys can be used (14.5.4).

12.3 Unsafe Software Management

The Premier system provides a number of software management features that allows users to update or otherwise “program” the system. These features lead to vulnerabilities that allow an attacker to compromise the system in serious and sometimes undetectable ways.

12.3.1 Installation Procedures

The AV-TSX automatically upgrades its software without any protections. Any memory card inserted into the AV-TSX while it is restarted is accepted as an authentic update. The AV-TSX loads files from the memory card to replace its bootloader and operating system (13.3.1) or Premier election software (13.3.2). These operations and errors in the installation process itself (13.3.3) allow an attacker to completely replace all software on the voting terminal. Note that no logging of the software upgrade occurs—once completed, there is no way of knowing the software has been replaced.

The ExpressPoll electronic poll book operates in essentially the same way—the bootloader and operating system (14.6.2) can be replaced by an adversary. These attacks could be used to monitor poll activity and possibly aid in recovering voter choices (13.3.19), or to add fraudulent voters or remove legitimate ones (14.6.3).

The Voter Card Encoder (VCE) loads whatever software the user provides it (14.5.3). The attacker restarts the device, accepts the potentially malicious software upgrade through the button interface, and the VCE reprograms itself with software provided by an attached laptop or other device.

12.3.2 AccuBasic

AccuBasic is an interpreted computer programming language that creates reports generated from the election data. The AccuBasic scripts (programs) are read from local memory cards by the AV-OS and AV-TSX

voting systems and executed on the local machine, for example, to print “zero-reports” at the beginning of an election day.

AccuBasic programs stored on the memory card can be used to compromise the AV-OS (13.2.10) or AV-TSX (13.3.13). There are design and code errors (e.g., buffer overflows) which can be used to compromise the voting device. Once compromised, the attacker can modify vote counts, ballot images, and forge reports without restriction. Moreover, improperly designed scripts can be used to print false reports or crash the machine entirely without exploiting any bug in the AccuBasic interpreter. These attacks can be masked in the AV-OS by exploiting a design flaw in the audit log design (13.2.4). A previous analysis of AccuBasic also identified deep flaws in its implementation, and highlighted a number of serious issues that arise from its use in elections.

12.4 Design and Code Quality Problems

Errors in coding and design are widespread in the Premier system. These issues are visible in the descriptions in the following chapters, and lead to serious vulnerabilities that can affect the processes and accuracy of an election. Recognition of these broad failures in the engineering of the Premier system is not new to this analysis—the underlying reasons for these issues have been widely reported. This section highlights some of the more important of these causes.

The issues found in the Premier system can most centrally be attributed to:

- *Complexity* - The Premier system is comprised of many components. The devices run on a variety of hardware platforms and run software built on many thousands of lines of source code developed in several different programming languages. Even under the best of circumstances, ensuring security in such a complex environment is exceptionally difficult.
- *Lack of Software Integrity* - The Premier system does not provide the basic mechanisms to ensure the integrity of software. As a result, it is prohibitively difficult to determine if the software running on the Premier system during an election has been compromised or replaced by an attacker.
- *Security and Software Engineering Practices* - Premier does not exhibit best security practices appropriate for high value systems. The apparent lack of detailed security requirements engineering and modeling, absence of security training, poor coding, failure to apply appropriate software testing and red-teaming has led to the vulnerabilities discussed in this report.
- *Reliance of COTS software* - Much of the Premier system is built on commodity software such as the Windows operating system. Such systems have many vulnerabilities that are exploitable by an attacker. Moreover, these vulnerabilities are uncovered constantly, thus presenting nearly insurmountable problems in defending the voting equipment from an attacker (who may know about an exploit long before a patch exists to fix it). Such problems are not limited to the operating system—there are several third-party commercial software libraries upon which the systems are built. The evaluation teams did not have access to the source code of these libraries.

One might be tempted to believe that coding or design errors lead to inconsequential vulnerabilities. Such a belief is unfounded—even small coding errors can expose the system to fatal misuse. For example, simple failures in design and coding result in issues that allow a voter to cast as many votes as they want (14.8.8) or allow a malicious election official to “pre-stuff” the ballot box before the polls open (13.2.7).

The remainder of this section identifies three areas of central concern, and demonstrates the how these failures lead to often serious system issues. These descriptions do not attempt to catalog all such issues, but highlight several the are emblematic of the root causes of the classes of failures.

12.4.1 Inadequate Security

One of the central limitations of the Premier system design is its failure to anticipate attacks. For example, removable memory card contents are unprotected from reading or modification in the AV-OS (13.2.1). This design introduces vulnerabilities that require draconian procedural controls over the card. Such controls prohibit such common practices as equipment sleep-overs. To cite a second example, a single AV-OS administrator password is used throughout a county. Therefore, anyone authorized to access any AV-OS in a county is implicitly authorized to access all AV-OS devices in the county. This is a failure of *least privilege*, an accepted principle of security design that mandates no user should be given more access than is strictly necessary. The same least privilege failure is manifest in the use of singular country-wide keys (14.1.2) and certificates (14.1.6).

Other design problems reflect a failure to understand required security functionality. For example, the audit log on the AV-OS allows up to 512 log entries (13.2.4). After the 512th log entry is created, the oldest log entry is overwritten. The purpose of an audit log is to be able to track the historical operation of the system. This functionality is essential to uncovering and understanding attacks, and for providing evidence of correct operation. The AV-OS log fails to meet this requirement in that it only understands a limited window of history. This leads to issues that can compromise or invalidate the audit function—any attacker who can force the system to create audit log entries (such as by causing errors) can cover his tracks.

Still other failures represent failures to embrace good security practices. For example, the very idea of a “default cryptographic key” is counter to good security (14.1.3). There is no such thing as a default secret, as under a reasonable security definition, if it is a default, then it is not a secret. Thus, by providing an insecure default the system behaves in an insecure way unless it is explicitly configured to do otherwise. Other examples of failures of good practice are the lack of effective authentication in software installation and update (13.3.1, 14.6.2) and the lack of safe programming practices (see below).

12.4.2 Improper Use of Security Technology

Improper use or implementation of security is the source of a large number of issues in the Premier system. Such problems are a consequence of a lack of security training for the designers and coders, and in many ways reflect an incomplete or flawed security model for the system.

One class of technology misuse is the improper implementation of cryptography. Standardized methods for ensuring integrity of data and software are not applied, e.g., digital signatures and hashed message authentication codes. Instead, home-brewed and often trivially breakable solutions are used. For example, check-summing (13.2.3) and other forgeable metrics (13.2.5) are used for integrity metrics (to determine if an attacker has corrupted data or software). Thus while their intent is laudable, the misunderstanding of common principles of security renders their function essentially useless. Other examples of misuse of security include, among many others, faux encryption (breakable obfuscation 13.2.8), poor security protocol implementation (13.2.2), and misuse of digital certificates (14.1.6).

A second class of technology misuse relates to a failure to appreciate the limitations of applied security technologies. To illustrate, much of the security of the back-end systems and data relies on the Windows

security infrastructure. Such controls have a long history of bugs, and there are many limitations of their use. In one such case, GEMS restricts access to key features by “graying out” (disabling) certain menu items on the user interface when logged-in users are not authorized to use them. However, disabled menu items in applications can be enabled using freely available Windows tools—thus, a user can simply circumvent the security mechanisms by re-enabling the menu items (13.1.3). Other mechanisms attempt to protect important data such as the election database using Windows filesystem access rights, but such protections again are easily circumventable (13.3.10). The Digital Guardian security tool attempts to augment the Windows security mechanisms, but flaws in its design and implementation allow it to be readily circumvented (14.7.1, 14.7.3, 14.7.4, see Section 12.5.2 below).

12.4.3 Unsafe Programming Practices

Attackers often compromise systems by exploiting simple coding errors. Many of the problems identified in this report are directly related to these errors. Our analysis shows that poor coding practices are visible throughout the system.² We illustrate the poor code quality by highlighting the substance and impact of two demonstrative classes of coding failures below:

- *Input checking* - A program that assumes but does not check that a particular input is *correct* can often be mislead, often critically, into crashing (14.1.4) or behaving in malicious ways (13.3.18). Common attacks that exploit improper inputs include buffer overflows (13.2.6), integer overflows (13.2.7), injection attacks (13.1.6), and `printf` attacks (13.2.6). These attacks can be used to infect the system with a virus, to change vote totals, to extract cryptographic keys and passwords, or any number of other malicious activities.

Note: Input checking errors are ubiquitous in the Premier system. We found many more than was feasible to report, and we are certain that there exist many more than we found. It seems unreasonable to expect that such errors will or can be completely eradicated from the source code, and thus they will continue to represent serious sources of security and reliability problems.

- *Misplaced Trust* - Programs that assume external entities or data sources are *trustworthy* are often subject to manipulation. For example, misplaced trust leads to incorrect software being installed (where every memory card and local filesystem is trusted to provide correct updates—a possibly unreasonable assumption, 13.3.2, 14.6.2). Note that misplaced trust is not just a coding issue—misplaced trust in the broader design can also lead to vulnerabilities that exploit falsely placed trust in the users or system components.

Ultimately, the Premier code exhibits a pervasive lack of *defensive programming*. To simplify, defensive programming is a style and practice of writing source code that goes to great lengths to identify and respond to bad or fraudulent data and environments. This includes, among many other strategies, checking for errors, validating inputs, failing gracefully when errors are detected, and auditing security relevant events in the system. Thus, a necessary component of improving the security of the Premier system is not only fixing the bugs in the system, but also changing the mindset of the developers as they fix them. A defensive philosophy must be part and parcel of future design and development efforts. Failure to embrace such philosophies will result in continued security and quality issues.

²The poor quality of the Premier system code has been widely reported. We do not provide a detailed treatise on secure programming in this report, but refer to numerous and extensive critiques in previous reports, programming texts, and scientific literature on software and security engineering for the problems of and solutions to poor code quality.

12.5 Previously Unreviewed Components

This study included four Premier components that had not been subject to prior analysis. We briefly characterize the findings of our evaluation of these components below.

12.5.1 EMP

The Election Media Processor (EMP) is stand-alone device that creates and reads up to six memory cards at a time. This device is used to make the process of setting up and tallying an election more efficient. EMP runs a Windows operating system. The vast majority of the EMP system was developed using source code from the AV-TSX touchscreen terminal (the only device previously available for creating or tallying memory cards).

The EMP appears to have imported several code quality issues from its AV-TSX predecessor. Stability issues are prevalent; poorly or maliciously formatted data from a AV-TSX (14.1.11), the GEMS server (14.1.10), or user input (14.1.4) can freeze or crash the EMP in dangerous ways. Such problems may allow an attacker to compromise the EMP software. Once compromised, the malicious EMP can then compromise other GEMS, AV-TSX, and AV-OS devices through virus-infected communications or memory cards.

Further security issues found in prior systems are also present in EMP; the cryptographic keys used to protect critical components are fixed—thus insecure—(14.1.10), can be poorly chosen (14.1.3), are widely distributed (14.1.2), or are manipulable (14.1.8). The impact of these issues is that an attacker who gains even brief access to the system is able to modify or other forge election data or introduce viruses that can compromise all election systems within the county (14.1.1).

12.5.2 Digital Guardian

The Digital Guardian (DG) software package is installed on the GEMS server to improve its security. This serves as a extra layer of protection against attackers who gain access to the computer and attempt to manipulate Premier and Windows files and applications. DG implements a local *policy* (configuration) that specifies which users may access files and databases, and which programs can and can not be used.

We found that DG was circumventable. The utility could be eliminated from running at all by booting the system from an external source (14.7.3), by circumventing boot protections (14.7.1) or by simply by disabling of the utility through the operating system management tools (14.7.4). These attacks can be used to render DG inoperable, and thus remove its protections.

DG's policy enforcement was also ineffective. The means by which dangerous applications are prevented from running are easily circumventable (14.7.6) or fail altogether (14.7.7). Moreover, the scope of such protections was very limited (14.7.9), and could be used to completely bypass the protections on, for example, elections databases (14.7.8).

12.5.3 ExpressPoll

ExpressPoll is used to register voters at a polling place, and is intended to replace the physical poll book. We were not given source-code to perform our analysis. Therefore, there may be errors in the implementation that we did not have resources to discover.

We were able to completely compromise the ExpressPoll within a few hours; however, once discovered, the attacks take seconds. The voter database is unprotected (14.6.3), leaving it vulnerable to an attacker. Further the unsafe software updates (14.6.2), booting procedures (14.6.2), and software configuration practices (14.6.4) render the ExpressPoll highly vulnerable to compromise. These vulnerabilities could be used to introduce a number of illegitimate voters into a precinct, or prevent legitimate voters from voting by removing them from the database.

Finally, note that while the ExpressPoll is currently not certified in Ohio, the ExpressPoll unit we received was from a precinct in Richland county.

12.5.4 Voter Card Encoder

The Voter Card Encoder is a pocket-calculator sized device used by poll workers to give voters access to cast their ballot on the AV-TSX. Before an election occurs, an election administrator inserts a valid Security smart card to program the device with the correct cryptographic keys. The election administrators at each precinct can then activate these devices by inserting their Supervisor smart card. Poll workers can then simply insert Voter smart cards and press “YES” to encode them.

This simple device acts as the gateway to an election; however, it is not protected as such. Once activated, we found no restrictions in number of valid voter cards that can be created. An attacker able to steal such a device could therefore cast multiple votes (14.5.2). Simply having access to the Voter Card Encoder is enough to load arbitrary software onto the device (14.5.3), an attack that is exploitable to make the use of a Supervisor card unnecessary. Moreover, this device accepts Supervisor smart cards with a default cryptographic key that has been posted on the Internet for years (14.5.4).

12.6 Audit Procedures

Audit logs are a necessary tool when performing a forensic examination of any computer. If a machine malfunctions or an attacker gains control, such logs can alert an auditor of the presence of such events. However, the ability to use audit logs is entirely dependent upon their accuracy. If an attacker can gain control of a machine and then edit logs or if legitimate machine failures are not recorded, the value of such logs are minimal at best.

All voting equipment created and used by Premier, except Voter Card Encoder devices, records audit logs in some fashion. However, sufficient mechanisms to protect the integrity of these devices are not present in these systems. Because of the key management issues in the AV-TSX (13.3.5), a virus can delete or fabricate logs and hide its presence. Potentially critical changes, such as loading a new bootloader or operating system (13.3.1) or BallotStation software (13.3.2) are simply not logged. The AV-OS can record a maximum of 512 operations in its audit log, after which it simply copies over old information (13.2.4). An attacker could simply compromise such a machine and perform 512 operations to cover their tracks. A less patient adversary could simply change the logs on the memory card as they are not protected with any key (13.2.1). The log files on the EMP server (14.1.5) and the ExpressPoll (14.6.6) are also not protected by any key allowing anyone with access to the system to delete or forge this information. The audit logs for the GEMS server, which are stored in the election database itself, can also be modified by anyone with access to the file (13.1.2). Digital Guardian, which has logging capabilities, does not record logs in Ohio (14.7.10).

Secondary sources of audit logs, such as VVPAT units, are also easily circumvented or disabled. Because the

plastic housing protecting this device is flexible, an attacker can easily cut the wires to the printer (14.8.1) or steal (14.8.2) the paper record. An attacker could also simply inject a common household chemical into the AV-TSX housing and, without breaking any seals, destroy all previously cast ballots (14.8.3).

The audit logs created by Premier voting equipment are therefore not sufficiently protected to provide trustworthy information for forensic analysis.

PREMIER ISSUE CONFIRMATION

The sections of this chapter detail our confirmation of the issues raised in the Diebold/Premier 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 non-public 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 conclusively confirm 41 of the 44 issues raised in the original California TTBR report and all 3 additional issues identified in the Connecticut study of the AV-OS PC. For two of the California TTBR report issues (CA issues 5.1.10 and 5.2.13), our inability of conclusive confirmation hinged only on a lack of access to unredacted California reports and extensive time required to reproduce the analysis. These issues were confirmed by two previous studies, and we have no reason to believe they are not correct. Of the remaining exceptions, CA issue 5.1.5 (13.2.5) indicates that the AV-OS PC memory card can be replaced while the ballot is being scanned; however it was previously only analyzed from the source code aspect. Our analysis of the source code came to the same conclusion as the CA review; however, we were unable to produce an environment under which we could remove the card without the system halting.

13.1 Premier GEMS Vulnerabilities

13.1.1 GEMS uses the Microsoft Jet data layer

The election database is stored in a single file interfaced by the Microsoft Jet data layer. This interface is the same used by Microsoft's Access office productivity software. Many question Jet's integrity under large concurrent loads. An attacker causing undue stress to the GEMS server during tabulation could cause general miscalculation of results.

Description: The GEMS server stores its election database, including information about races and post election results, in a Microsoft Access .mdb file using the Jet data layer. While this provides convenience when backing up an election by burning a single file to CD-R, many question the integrity of Jet, and

Microsoft itself recommended against using Jet in Cuyahoga County, Ohio.¹ Microsoft recommends using MS SQL server for critical database applications. During our investigation of the GEMS, we found that the source code appears to contain support for MS SQL server. Hence, moving to an architecture that uses MS SQL server may not require significant changes to the GEMS source code.

This issue was previously identified by Ryan and Hoke,² and reiterated by the CA Diebold source code review (5.3.1). Ryan and Hoke provide a more detailed look at the use of Jet.

Prerequisites: Participation in the tabulation procedure.

Impact: By using Jet, the GEMS tabulation is subject to potential corruption. Scott Massey, a Microsoft Spokesman, indicated that connection loss during data transfer can result in database corruption. An attacker causing undue stress to the GEMS system or connected network during the tabulation process could cause general miscalculation of results.

Procedural Mitigations: None.

Verification: This issue was confirmed through source code analysis and study of documentation.

13.1.2 GEMS databases can be modified with access to the local hard disk

When an election database is opened, GEMS prompts the user for a username and password and restricts what parts of the election database the user can read and modify. An attacker with access to the local hard disk can directly read or modify the election database to overcome these restrictions. This technique can be used to obtain passwords, or change the election setup after it has been finalized.

Description: Every election database contains a list of valid usernames and passwords. GEMS uses this list to ensure that only authorized users can view and make modifications to an election. Additionally, once a database has been “set-for-election,” no one should be allowed to make changes to the election, e.g., modifying the candidates on the official ballot.

While the GEMS application can restrict how users access the election database through it, it cannot control users who access the database via other means. GEMS stores each election database in a single .mdb data file; this is the same format used by Microsoft Access. We confirmed that election databases can be opened and modified by Microsoft Access as well as other file (HEX) editors.

Modifying the election database is easier than described in the CA Diebold source code review report. As a proof of concept, we wrote a five line Visual Basic script using the ADODB library to retrieve the election administrator’s password hash (see Issue 13.1.8). The script can be written in any text editor from an attacker’s memory in a matter of minutes and can be trivially modified to perform any SELECT, UPDATE, and INSERT query. Additionally, we created a slightly longer script (under 50 lines) that provides an attacker an interactive SQL shell. Such a tool would allow an attacker to freely explore the election database and destroy tamper evidence with minimal prior knowledge.

This issue was previously identified in the CA Diebold source code review (5.3.2).

Prerequisites: To exploit this vulnerability, an attacker requires a few minutes access to the GEMS server file system. In particular, the .mdb files used to store election information and results.

¹Bob Driehaus, ‘Audit Finds Many Faults in Cleveland’s ‘06 Voting’. The New York Times, April 2007 (URL: <http://www.nytimes.com/2007/04/20/us/20ohio.html>), ISSN 0362-4331.

²Thomas Ryan and Candice Hoke, ‘GEMS Tabulation Database Design Issues in Relation to Voting Systems Certification Standards’. In Proceedings of the USENIX/ACCURATE Electronic Voting Technology Workshop. August 2007.

Impact: An attacker has complete access to the election database to read and make changes to any information. Example modifications include changing a ballot definition after a database has been “set-for-election” (similar to Issue 13.1.5), manipulating the audit log to hide tampering, and creating new database users with administrative access (see Issue 13.1.8). User passwords can also be derived from the database contents.

Procedural Mitigations: Prevent unauthorized access to .mdb files. Note, however, that Chapter 14.7 describes vulnerabilities in Digital Guardian, a security tool added to protect these files.

Verification: This issue was confirmed via demonstration with the GEMS server.

13.1.3 GEMS relies on the graphical user interface (GUI) to enforce security

The GEMS application keeps users from performing certain actions, e.g., performing unauthorized administrator actions or modifying an election definition after it has been “set-for-election”, by disabling (“graying out”) parts of the interface to make them inaccessible. Freely available software used simply to supplement basic Windows features can enable any interface in windows. This software can be used to subvert the GEMS controls.

Description: Each user of the GEMS server has limited access to the database. The practice of least privilege, or granting each user only the rights they need to do their job and none more, is an excellent security practice. Unfortunately, the mechanism by which least privilege is implemented on GEMS is vulnerable to attack.

Access to the database is controlled through the use of “widget” state. For instance, user *Alice* can use GEMS to send ballot definitions to the EMP server. *Alice* can not, however, access administrative options or change fields such as party affiliations of candidates. To prevent access to these fields, GEMS grays out these options. However, the access control mechanism does not extend any further than the GUI.

Using a freely available program, we were able to access all administrative options as the *Alice* user. Every field in the database could be changed even if access was not originally given to a particular user. The property of least privilege was therefore violated and control of the ballot given to the attacker.

In addition to providing privilege separation, GEMS restricts certain actions to take place once a database has been “set-for-election.” Using this freely available program, we were able to change various parts of the election database after memory cards had been created. Of most significance, we were able to modify fields that control the election reporting sequence. This makes Issue 13.1.5 significantly worse.

This confirms and extends the vulnerability (5.3.3) reported in the California Source Code Report.

Prerequisites: Anyone with an account on the GEMS server, the ability to copy files to the GEMS server, and an account for the target election database could launch this attack.

Impact: An attacker could change any field in the database, potentially corrupting the ballot before it was sent to polling sites. Election results could also be corrupted by switching the mapping between candidate names and identifiers. With unrestricted access to GEMS and the database, there are few limits to what an attacker could achieve.

Procedural Mitigations: None.

Verification: This issue was confirmed via demonstration with the GEMS server.

13.1.4 Third parties translation services may introduce a virus into the system

The Premier system documentation referring to translation agencies instructs the election administrator to either send the entire election database or a few configuration files to a translation agency and to consider these files as official upon return. An attacker at the translation agency could use these files to introduce a virus in to the system. However, to our knowledge, Ohio does not use third party translation agencies.

Description: Both GEMS and the AV-TSX use RTF files to provide multiple language support. Issue 5.3.4 of the CA Diebold source code review indicated that the “GEMS Election Administrator’s Guide” described two processes for translating English into a foreign language. One process (Section 5.4.2.3) includes sending the entire election database to the translator, and the returned database is considered the official database. A malicious translation agency can make undetected changes to the ballot definition and introduce changes that exploit the GEMS server (see Issue 13.1.7). The other process (Section 5.4.2.2) uses export/import RTF features of GEMS to provide the translator only with the RTF files. While safer, the latter process still allows the translator to create malicious RTF files that may exploit buffer overflows in the GEMS and AV-TSX (see Issue 13.3.15).

The CA source code review team was provided revision 8.0 of the “GEMS Election Administrator’s Guide,” which was published June 2005. We were provided revision 10.0,³ published May 2006. Note that revision 9.0 was published less than a month after revision 8.0 and contains a very long changelog indicating conformance with GEMS 1.18.24. Revision 10.0 contains a terse changelog indicating: “Entire document re-written.” The sections referenced in the CA report are no longer valid in revision 10.0 of the document. The document rewrite appears to have rearranged chapters, shifting sections 5.4.2.2 and 5.4.2.3 to 4.4.2.2 and 4.4.2.3, respectively. These sections capture the same notions of trust in the translation agency as reported by the CA report.

This issue was previously identified in the CA Diebold source code review (5.3.4),

Prerequisites: Access to files at the translation agency.

Impact: Malicious RTF files and election database content can exploit GEMS and AV-TSX software to introduce a virus. However, to our knowledge, Ohio does not use third party translation agencies.

Procedural Mitigations: As a *partial* mitigation, only the Export/Import RTF procedure should be used. Before importing the RTF files, a security audit of the returned RTF files should carefully look for malicious modification.

Verification: This issue was confirmed through review of available documentation.

13.1.5 Race and candidate labels may be changed after GEMS has been “set-for-election”

GEMS allows users to fix spelling mistakes in the race and candidate labels after an election has started. An attacker can “fix” the spelling by swapping candidate and race names.

Description: The GEMS election database uses standard schema design practices for storing and referring to user created information. As such, candidates and races are not identified by their textual name but rather a fixed integer, known as a *primary key* in database jargon. All references to a candidate or race, including vote counters, use the primary key. This design is well accepted and appropriate for database applications such as GEMS.

³Diebold Election Systems, *GEMS 1.18 Election Administrators Guide, Revision 10.0*. May 15, 2006.

Unfortunately, a design decision was made to allow race and candidate labels (but not party affiliation) to be changed after the system has been “set-for-election.” Presumably this feature exists to fix spelling mistakes. However, it has the unfortunate potential side-effect of allowing a GEMS user to redistribute vote totals, even without the tool described in Issue 13.1.3.

The CA Diebold source code review described a scenario in which two races are swapped. In the first race (Mayor), the republican candidate won, and in the second race (District Attorney), the democrat candidate won. By swapping the race and candidate names, the election results indicate that the democrat candidate won the Mayor race and the republican candidate won the District Attorney race.

This attack exploits the mapping between textual label and database primary key. That is, vote counters refer to the integer primary key, and if the attacker changes the textual name associated with that integer, the results summary will happily display the new text. However, in order to exploit this vulnerability, an attacker must change the database used to generate the results summary. For ease of reporting, the Premier system includes the concept of a “master GEMS server” that can reside in the Secretary of State’s office. The results from all counties using Premier equipment are uploaded to the master server, which tallies votes across counties and displays results. Any changes to the candidate and race strings at the county level will not affect the state level report. Hence, to exploit this vulnerability, the attacker must have access to the master GEMS server. However, to our knowledge, Ohio does not use a master GEMS server, therefore the attack can be performed at the county level.

The CA Diebold source code review report also indicated that an observant election reviewer could detect the tampering by comparing the race order in the election summary. In the above example, the race for District Attorney appears before the race for Mayor. Additionally, the change can be detected upon close investigation of the election database in GEMS. However, most if not all of these traces can be removed using the tool described in Issue 13.1.3.

This issue was previously identified in the CA Diebold source code review (5.3.5)

Prerequisites: A GEMS account on the GEMS server used to create the official election results.

Impact: The candidates of two races can be swapped, possibly changing the winner of each race.

Procedural Mitigations: An election summary report should be produced before an election and the ordering of races should be carefully compared to the final election summary. Additionally, summary tapes of all polling places should be manually compared against the official election results.

Verification: This issue was confirmed through source code analysis and via demonstration with the GEMS server.

13.1.6 GEMS fails to filter login field for special characters

An attacker with a valid username and password to an election database can exploit flaws in the GEMS server login process to perform arbitrary database lookups, including finding the password of other users with administrative access.

Description: Each election database has an associated list of users that have access. Users can optionally have administrative access to the database, which provides extra privileges, e.g., creating new users. General security practices recommend each system user has a unique username and password, thereby allowing audit logs to include accountability.

The user opening a GEMS election database is prompted with a dialog box asking for a username and password. GEMS does not perform any filtering on the username field. This allows an attacker to perform an “SQL injection attack” by escaping the database query using the `'` character.

The CA Diebold source code review team discovered a significant limitation of this attack. They found that vulnerability can only be used to perform additional `SELECT` queries on the database, and not `INSERT` and `UPDATE` queries, which modify the database. They believe this limitation is a result of the Microsoft Jet data layer (see Issue 13.1.1, which recommends against the use of Jet) and that if Microsoft SQL server were used instead, the `INSERT` and `UPDATE` queries would be allowed. As GEMS is configured to only work with Jet, we could not confirm if using Microsoft SQL server would allow these queries.

The ability to execute arbitrary `SELECT` queries is damaging in itself. The CA Diebold source code review team showed how the login field could be used to query for the existence of other users and whether the user has administrative access. Additionally, due to the way GEMS stores users passwords (Issue 13.1.8), an attacker can determine all of the usernames and passwords for an election database. This information may be used to gain administrative privileges or even cause malicious database accesses to be logged a different user.

This issue was previously identified in the CA Diebold source code review (5.3.6).

Prerequisites: Anyone with a GEMS server and election database account can mount this attack.

Impact: This vulnerability allows an attacker to retrieve arbitrary information from the election database, including information about other election database users.

Procedural Mitigations: None.

Verification: This issue was confirmed through source code analysis and via demonstration with the GEMS server.

13.1.7 Unsafe handling of election database content may allow a virus to control GEMS

GEMS trusts that data in the election database is malformed, and therefore does not always provide necessary input filtering. An attacker may modify the election database outside of GEMS to exploit this lack of filtering and insert a virus into the system.

Description: In many cases throughout the source code, GEMS trusts that the data returned from the database is not malformed. However, there is no guarantee that the information within the database has not been altered, maliciously or otherwise (Issue 13.1.2). Accordingly, there are a large number of instances throughout GEMS in which buffer overflow attacks are possible. We discuss two below:

One function in GEMS takes an arbitrarily long `CString` as input and returns the same string minus zeros and whitespaces. As the function iterates through the input character by character, it places non-zero and non-space characters into a buffer. The buffer, however, is initialized to hold no more than 128 characters. Accordingly, an attacker can cause a buffer overflow if they alter a field in the database to contain more than 128 characters. This vulnerability was mentioned in the California Diebold Source Code Report.

Another function in GEMS is responsible for queueing jobs. This function takes a filename and a “friendly”⁴ filename from the database. Both of these names are of the type `CString`, meaning that each string can hold an arbitrarily long value. Both of these values are placed into a fixed size buffer using `sprintf()`.

⁴A “friendly” file name is more human-readable and can have spaces.

Accordingly, this code is susceptible to a buffer overflow. This particular vulnerability was not previously mentioned in the California Source Code Report.

Such problems are common throughout the code.

This confirms the class of vulnerabilities (5.3.7) reported in the California Source Code Report.

Prerequisites: An attacker requires access to the GEMS database in order to exploit this vulnerability.

Impact: An attacker could use these buffer overflows to gain control over GEMS and run arbitrary software or introduce a virus. The integrity of the election could be compromised by the exploitation of only one such vulnerability.

Procedural Mitigations: None.

Verification: This vulnerability was confirmed through source code analysis.

13.1.8 Election database passwords are stored with insufficient protection

GEMS stores election database user passwords in the database using a known weak algorithm. An attacker with read access to the database can retrieve the stored values to derive all user passwords. Additionally, an attacker with write access to the database can change any user's password or create a new user.

Description: GEMS controls access to election databases by usernames and passwords. The password is hashed using the `DES_crypt()` OpenSSL function and a two character salt (created with `rand()`, initialized by the current time). This is similar to the `/etc/passwd` file in early UNIX systems. An attacker who retrieves the hash value can use widely available dictionary attack software to determine the password. Additionally, since GEMS authentication simply looks up a username in a database table, an attacker can simply add a new entry to trivially create a new user.

The vulnerability becomes worse when combined with issue 13.1.6, in which any valid user can execute arbitrary `SELECT` queries. Using Issue 13.1.6, an attacker with a valid username and password can extract the password hash of all other users using a series of carefully constructed queries. The attacker can then run widely available dictionary attack software against the hash.

We confirmed that password hash extraction is possible via SQL queries. Using a binary search technique (suggested by the CA Diebold source code review report), we determined that an attacker requires at most 91 queries to extract a password hash. This accounts for the 13 character output of `DES_crypt()` (including the salt) and performing additional queries to determine the character case (all SQL queries in the current configuration are case insensitive). Such an attack would only take a matter of minutes.

This issue was previously found by the CA Diebold Source code review (bug:5-3-8).

Prerequisites: Extracting the password requires the attacker to have either a valid login to the election database or read access for the `.mdb` database file. Adding an additional user requires that the attacker have write access to the `.mdb` database file. Deriving the password from the hash may take hours or days, depending on complexity of the password, depending on complexity of the password.

Impact: An attacker with local access can create new accounts to control elections in under 30 seconds.

Procedural Mitigations: None.

Verification: This issue was verified using manual source code analysis and testing with the GEMS server.

13.1.9 Poor handling of integers may cause GEMS to crash

When converting integers to strings, GEMS writes the converted values into buffers that are too short. While the amount that the buffer is overflowed is likely too small to run arbitrary code, an attacker may still be able to cause GEMS to crash.

Description: GEMS copies signed integer values into fixed-length buffers. Declared to hold a maximum of 10 characters, these buffers are too short to represent the entire range of possible numbers that can be input. Because a 32-bit integer can contain up to 10 digits, the string representation may require an additional two bytes to for a negative sign and the string terminator (“\0”).

Because the amount that the buffer is overfilled is so small, it is unlikely that this vulnerability can be used to run arbitrary code on GEMS. However, an attacker may still be able to cause GEMS to crash.

This confirms the vulnerability (5.3.9) reported in the California Source Code Report.

Prerequisites: An attacker requires write access to the election database for a short amount of time to exploit this vulnerability. Depending on the field, this attack may be exploitable from within the GEMS application.

Impact: An attacker could crash the GEMS server.

Procedural Mitigations: None.

Verification: This vulnerability was confirmed through source code analysis.

13.2 Premier AV-OS PC Vulnerabilities

13.2.1 The AV-OS memory card data is not authenticated

The AV-OS PC memory card contains all of the information about an election, including the ballot definition and the tallied election results. Because there are no cryptographic protections for data on the memory card, an attacker could modify various values to control an election and the results.

Description: The AV-OS PC memory card contains the election information required to process ballots and stores results for post-election processing. More specifically, it contains:

- Memory card header: firmware revision number, card size, election status, count mode (e.g., absentee), and global counters (for total ballots counted, overrides, number of election uploads, etc.)
- Election header: voting center name and number, download version number, obfuscated supervisor password (an unsigned integer), election title and date, election type, party code table, election configuration flags, and data checksums.
- Election definition: precincts, ballot cards, races, candidates, and voting positions
- Election data: race and candidate counters
- Audit log
- AccuBasic scripts (compiled)

- **Memory card heap:** Most of the above data is stored on the memory card heap and referenced using pointers in fixed positions.

Nothing prevents an attacker from maliciously modifying the contents of the memory card. Simple checksums are used; however, not only can checksum values also be rewritten, but it is trivial to replace values without needing to change the checksums (see Issues 13.2.3 and 13.2.9). Additionally, the security sensitive supervisor password is obfuscated, but trivially recoverable (see Issue 13.2.8).

This issue was previously identified in the CA Diebold source code review (5.1.1), however others have looked at changing the memory card contents for red teaming purposes.⁵

Prerequisites: To exploit this vulnerability, an attacker requires write access to the memory card.

Impact: The lack of memory card authentication is a catalyst for many of the AV-OS PC vulnerabilities. It allows an attacker to write anything to the memory card and entirely change the outcome of an election.

Procedural Mitigations: Tamper evident seals are a potential *partial* mitigation to denote access to the memory card slot has occurred. For practical limitations, see Section 3.5.

Verification: This issue was confirmed through source code analysis.

13.2.2 The GEMS server and AV-OS PC communication is not authenticated

The AV-OS PC communicates with the GEMS server to download the election definition and to upload election results. When downloading an election definition, the AV-OS PC does not ensure the GEMS server is legitimate. Hence, an attacker could pretend to be the GEMS server to define an incorrect ballot definition. On election results upload, the GEMS server does not ensure the AV-OS PC is legitimate. Hence, an attacker could upload incorrect election results.

Description: Upon inserting an empty memory card into the AV-OS PC, the user is prompted to initialize the card by downloading information from the GEMS server. The download can be performed either in direct mode (over a serial cable) or using a modem (however, modems are prohibited by Ohio law). Neither the AV-OS PC nor the GEMS server requires any form of authentication token, e.g., a password, and the communication occurs without encryption. The connection establishment uses a primitive challenge response protocol; however, the “passwords” and “keys” (indicated by variable names) are simple 16-bit integers, and the encryption function is the same simple mathematical function described in Issue 13.2.8. Furthermore, even if the protocol used reasonable sized keys and strong cryptographic ciphers, e.g., AES, it would not provide secure authentication.

After an election, the AV-OS PC is placed in “post-election” mode. At this time, the AV-OS PC can upload election results to the GEMS server. Like the download process, the upload does not use encryption or authentication.

This issue was previously identified by the CA Diebold source code review (5.1.2).

Prerequisites: A new election can be downloaded in a matter of minutes. To download an election to the AV-OS PC, the attacker must have a direct connection to the AV-OS PC serial port and the current memory card must be blank. However, if the memory card is not blank, obtaining the supervisor password (see Issue 13.2.8) allows the attacker to blank the card.

⁵Harri Hursti, *The Black Box Report: Critical Security Issues with Diebold Optical Scan Design*. Black Box Voting, July 4, 2005 (URL: <http://www.blackboxvoting.org/BBVreport.pdf>).

To upload election results to the GEMS, the attacker must have a direct connection to the GEMS server (or via proxy by inserting a forged memory card into the batch sent to the central office), and the memory card ID must have not already been used.

Finally, if modem communication is used, the attacker can perform either attack simply by hijacking the connection. However, Ohio law prohibits the use of modems with electronic election equipment.

Impact: An attacker could cause the AV-OS PC to make memory cards that “pre-stuff” an election (see Issues 13.2.9, 13.2.10, and 13.2.11). An attacker could also report incorrect election results to the GEMS server either via direct communication or by placing a forged memory card in the batch returned to the central office. However, the forged card must have a valid memory card ID that has not yet been used. If the GEMS server receives a duplicate ID, it will not count the second card. This effect could also be used to delay election result reporting, as it will require ballots to be recounted.

Procedural Mitigations: As the memory cards with the election definitions should always come from the central office, the modem and serial ports should be physically disabled in a way verifiable without opening the AV-OS PC, e.g., short all of the pins.

Verification: This issue was confirmed through source code analysis.

13.2.3 The memory card checksums do not protect against malicious tampering

The memory cards attempt to protect vote counts using a technique called “checksumming.” Checksums are a means of ensuring that accidental changes (errors in network transmissions or hard disks) are not detected. Because anyone can create a valid checksum for any data, malicious changes to data can not be prevented or detected. Furthermore, due to the checksumming method used on the AV-OS PC, an attacker can often change values without changing the checksum value.

Description: The memory card uses checksums to ensure data integrity. First, there is no way to ensure the checksum values, also stored on the memory card, have not been modified. Second, even if the checksums were protected, the checksum algorithm by the AV-OS PC is trivially overcome. Integer values are checksummed by adding the values. For example, as long as the total votes cast for a race remains the same, the vote totals for each candidate can be modified without detection. The character checksum is slightly more complex, as values are XORed with a counter. However, this algorithm is equally susceptible to malicious tampering. Finally, as the checksums are calculated *after* the memory card contents are downloaded from the GEMS server, they do not even protect the system from transmission corruption (malicious or otherwise).

This issue was previously identified by the CA Diebold source code review (5.1.3).

Prerequisites: In order to exploit this issue, an attacker requires the ability to maliciously modify values on the memory card.

Impact: Memory card contents can be modified even without changing the checksum values. The adversary merely needs to ensure the total sum does not change. That is, if one value is lowered, another must be raised. Such manipulation requires insignificant sophistication. This vulnerability is an enabler for more sophisticated memory card attacks to go unnoticed.

Procedural Mitigations: Tamper seals are a potential *partial* mitigation to protect memory cards. For practical limitations see Section 3.5.

Verification: This issue was confirmed through source code analysis.

13.2.4 The audit log is unprotected and old entries are overwritten

The audit log, a security feature used to detect abnormalities, is stored unprotected on the memory card. Due to space limitations, once the audit log fills, it begins overwriting the oldest entries. Not only could an attacker arbitrarily add, modify, and remove log entries, causing the machine to log events could remove all traces of an attack.

Description: The AV-OS PC uses an audit log to record major events and abnormalities. These events are recorded in an audit log on the memory card, which is viewed as a security measure to detect misuse. The event record storage is fixed at 512 entries and when the log runs out of room, the oldest entries are overwritten. Furthermore, there is no integrity check of the log, as described in Issues 13.2.1 and 13.2.3.

This issue was previously identified by the CA Diebold source code review (5.1.4).

Prerequisites: The audit log is appended during normal operation. To overwrite older events, which may reveal tampering, and attacker need only cause the device to report abnormal, but benign events. This may take one to two hours to perform. For the manual manipulation, an attacker would need access to write to the memory card.

Impact: This vulnerability allows an attacker to cover all traces of tampering with the device and/or memory card. It can also be used by a malicious AccuBasic script (see Issue 13.2.10) to hide events corresponding to an attack.

Procedural Mitigations: Tamper seals are a potential *partial* mitigation to protect memory cards. For practical limitations see Section 3.5. However, tamper seals cannot prevent old logs from being overwritten.

Verification: This issue was confirmed through source code analysis.

13.2.5 The memory card “signature” does not prevent malicious tampering

The source code indicates that the memory card contains a variable called a “signature.” While one may initially think this variable is a cryptographic signature, it is not. The variable is used to detect if the card was replaced during the scanning operation and does provide any cryptographic protection.

Our evaluation *cannot conclusively confirm or deny the issue*. Our analysis of the source code leads us to the same logical conclusion as the CA report; however, we were unable to recreate an environment to confirm the issue via demonstration.

Description: The AV-OS PC source code contains the structure `MemCardSignature` and a variable of this type stored on the memory card. The signature is recorded just before reading the ballot and checked immediately after returning. If the signature does not match, the machine halts.

The “signature” does not provide cryptographic protection. It consists of three counters: the total number of votes thus far, the number of nonabsentee votes, and the number of absentee votes. It does not protect any other values on the card.

The CA Diebold source code review identified this issue; however, they were unable to experiment with the equipment. Given that the AV-OS PC runs a single threaded operating system, our interpretation of the source code was that all the logic could not detect if the memory card was replaced during the ballot reading, as processing time is devoted to driving the motors and reading the timing marks. However, in experimenting with the AV-OS PC, we were unable to produce an environment that allowed the memory

card to be replaced during the ballot scan operation. In our of our attempts, the system halted as soon as the memory card was removed. As we discovered this late in our analysis, we were unable to further analyze the situation; however, we suspect the situation relates to system interrupts.

This issue was previously identified in the CA Diebold source code review (5.1.5).

Prerequisites: This vulnerability requires access to the memory card during the ballot scan operation. It also requires knowledge of the number of votes cast on the memory card up until the time of the attack; however, this is displayed on the AV-OS PC LCD.

Impact: As long as the total number of votes match, the memory card can be replaced during the scan operation.

Procedural Mitigations: Tamper seals are a potential *partial* mitigation to protect memory cards. For practical limitations see Section 3.5.

Verification: This issue was confirmed through source code analysis; however, we were unable to recreate an environment to confirm it via demonstration.

13.2.6 Improper string handling can result in arbitrary code execution

The memory card contains a number of textual strings to describe an election. The AV-OS PC assumes these strings are of a certain length and blindly uses them when communicating with the GEMS server. Due to the lack of string length checking, an attacker controlling the memory card could craft strings that cause the AV-OS PC to insert code into critical parts of the system, thereby allowing the attacker to take control.

Description: The source code that uploads election results from the AV-OS PC memory card to the GEMS server contains flaws that could allow an attacker to take control of the machine. The `sprintf()` function is misused in at least four places such that it may be exploitable if an attacker specially crafts fields on the memory card. One of these vulnerabilities allows the attacker to take control in enough time to dictate the entire reported election results.

The recommended replacement for `sprintf()` is `snprintf()`, which forces the programmer to specify the length of the buffer. The AV-OS PC source code does not currently have the ability to use `snprintf()`, as it is not defined in the custom IO library used for the device.

This issue was previously identified in the CA Diebold source code review (5.1.6)

Prerequisites: To exploit this vulnerability, an attacker needs to control the data written to the memory card. This may be from GEMS or by directly writing to the card.

Impact: A malicious memory card can result in arbitrary code execution that can control the election results reported to GEMS.

Procedural Mitigations: None.

Verification: This issue was confirmed through source code analysis.

13.2.7 Candidate vote counters are not checked for overflow

Computers store numbers in fixed size storage. When a number grows to large, it wraps around back to zero (a.k.a., overflow). The AV-OS PC does check to see if vote counters overflow. This allows an attacker to

start a candidate out at a disadvantage by initially assigning a very large number with the expectation that enough votes will occur to result in an overflow.

Description: The AV-OS PC uses 16-bit unsigned integers to store counters that track votes for each candidate. This means there is a maximum count of 65,535 for each candidate. When a 16-bit unsigned integer of value 65,535 is incremented, it takes the representation of 0. There is no checking to see if the counter overflows, therefore, an attacker can “pre-stuff” an election to give one candidate an advantage. See Issue 13.2.11 for an example.

This issue was previously identified in the CA Diebold source code review (5.1.7). A related attack was first demonstrated by Hursti.⁶

Prerequisites: To exploit this vulnerability, an attacker requires the ability to write to the memory card. A successful attack also requires other vulnerabilities for “cover-up.” See Issue 13.2.11.

Impact: The lack of overflow checking is an enabler for more sophisticated attacks. See Issue 13.2.11 for a description of how this vulnerability can be used to allow an attacker to create a scenario where one candidate is at a disadvantage at the start of the election. This issue also has the unfortunate side-effect that if a candidate legitimately receives more than 65,535 votes on one AV-OS PC machine, the total will be reset without detection.

Procedural Mitigations: Tamper seals are a potential *partial* mitigation to protect memory cards. For practical limitations see Section 3.5. Additionally, one memory card should never be used for more than 65,535 votes.

Verification: This issue was confirmed through source code analysis.

13.2.8 The supervisor “password” is stored with inadequate protection

An attacker with access to the memory card can discover the supervisor password. The password is stored in an obfuscated form; however an attacker could trivially derive the real value.

Description: The supervisor “password” is a simple integer (like a PIN) required to perform supervisor operations on the device. Such operations include setting the GEMS dial-up phone number, changing feeding options, configuring communications ports, changing the election status, duplicating memory cards, and clearing memory cards. The password is stored in obfuscated form on the memory card. The obfuscation routine combines the password with a key (another integer stored in cleartext on the memory card right next to the obfuscated password) and two 20-bit “magic” constants hardcoded in the source code. The obfuscated form *obf* of the supervisor password (PIN) is as follows: $[obf = encode(pass, key) = pass + (key \times MAGIC_1) + MAGIC_2]$ To recover the password, an attacker simply needs to compute: $[pass = decode(obf, key) = obf - (key \times MAGIC_1) - MAGIC_2]$ This algorithm appeared in the CA review and is trivial to reverse-engineer given a few minutes access to the machine.

Obfuscation is never a substitution for cryptographic protection. However, simply applying cryptographic protection, e.g., storing a cryptographic hash of the password, will not fix this vulnerability. Once the attacker acquires the hash, it would take under a second to brute force guess the password on even low powered devices such as a PDA.

This issue was originally identified by Kiayias et al.⁷ (who did not have source code) and confirmed by the

⁶Hursti, ‘The Black Box Report: Critical Security Issues with Diebold Optical Scan Design’ (as in n. 5).

⁷A. Kiayias et al., *Security Assessment of the Diebold Optical Scan Voting Terminal*. UConn Voting Technology Re-

CA Diebold source code review (5.1.8).

Prerequisites: To recover the password, an attacker requires read access to the memory card, e.g., reading the card directly or from the serial port in diagnostics mode (see Issue 13.2.14). The password is stored near the beginning of the card, therefore the entire card does not need to be read. Using diagnostics mode and the serial port, the obfuscated password and key can be retrieved in under a minute. If the magic values are known, the password can be calculated in microseconds.

Impact: This vulnerability allows anyone with access to the memory card to recover the password and perform special supervisor functions such as reopening an election after it has closed. Furthermore, the supervisor password is set in the “AccuVote-OS” dialog box of GEMS and cannot be modified after the system as been “set-for-election.” Therefore all optical scan units within a county use the same password.

Procedural Mitigations: Tamper seals do not completely mitigate this vulnerability, because all AV-OS PC passwords within a county are the same (GEMS has one text box for the password that is “locked” after the database has been “set-for-election”), an attacker can steal the password from one precinct and use it in another.

Verification: This issue was confirmed through source code analysis.

13.2.9 Candidate ballot coordinates can be modified to manipulate election results

Each election candidate has a defined coordinate for an oval on the ballot. An attacker could change the coordinates in the ballot definition on the memory card such that votes are swapped or the oval location is somewhere a voter will never fill in.

Description: The optical scanner detects a vote for a candidate by looking for a filled in oval at a specific coordinate on the ballot (the ballot is viewed as a grid with row and column coordinates). The vote position coordinates are downloaded to the memory card before an election. The only coordinate verification that occurs before an election is the simple checksumming algorithm described in Issue 13.2.3. As long as the sum of the row/column start/end values is equal to the original sum, an attacker can make changes without detection. For example, $(row, column)$ can be changed to $(row - 1, column + 1)$, without changing the sum. As the checksum operates over *all* candidates, candidate coordinates can be swapped without detection.

This issue was first discovered by Kiayias et al.⁸ and confirmed by the CA Diebold source code (5.1.9) and red team reviews.

Prerequisites: To exploit this vulnerability, an attacker requires the ability to write to the memory card.

Impact: An attacker could either swap candidates on a ballot, or shift a candidate’s vote position to an area where a mark will never exist, thereby ensuring no votes will be recorded. Both attacks could directly impact the election outcome.

Procedural Mitigations: Tamper seals are a potential *partial* mitigation to protect memory cards. For practical limitations see Section 3.5.

Verification: This issue was confirmed through source code analysis.

search (VoTeR) Center, October 30, 2006 (URL: http://voter.engr.uconn.edu/voter/OS-Report_files/uconn-report-os.pdf).

⁸Kiayias et al., ‘Security Assessment of the Diebold Optical Scan Voting Terminal’ (as in n. 7).

13.2.10 Flaws in the AccuBasic interpreter may allow a virus to control an AV-OS PC

The memory cards contain “live code” that is run by a special interpreter on the AV-OS PC. This interpreter contains flaws that may compromise the machine’s integrity. As the “live code” comes from the GEMS server, this vulnerability provides potential for viral spread.

Our evaluation has not conclusively confirmed or denied this issue. Due to time limitations and lack of access to specific unredacted reports, we were unable to identify the vulnerabilities found in the previous studies. However, we have no reason to believe the vulnerabilities do not exist.

Description: The AV-OS PC firmware resides on a read-only hardware chip. To provide flexibility and customization, AccuBasic scripts are placed on the memory cards. These are “live” pieces of code that are executed by the AccuBasic interpreter and perform customized functionality such as printing the Election Zero report.

There are a number of flaws in the way the interpreter handles script contents, e.g., buffer overflows, that allow a script to take over control of the machine. A virus infected GEMS server can create malicious AccuBasic scripts that infect the machine, e.g., it will infect the machine when the supervisor runs the Election Zero report.

This vulnerability was originally discovered by Wagner et al.⁹ in an extensive study of the AccuBasic interpreter for the California Secretary of State and subsequently confirmed by the CA Diebold source code review (5.1.10). See Wagner et al. for a detailed description.

Prerequisites: An attacker requires the ability to control the AccuBasic scripts written to the memory card. This could be done either by controlling the GEMS server or by gaining write access directly to the memory card.

Impact: This vulnerability allows an attacker to escape the restricted control of the AccuBasic interpreter, providing the ability to write directly to vote counters and other critical storage. Most importantly, it allows a virus infected GEMS to extend to the AV-OS PC.

Procedural Mitigations: None.

Verification: Our evaluation has not conclusively confirmed or denied this issue. Due to time limitations and lack of access to specific unredacted reports, we were unable to identify the vulnerabilities found in the previous studies. Because this issue has been reported by Wagner et al and then verified in the California report, we have no reason to believe the vulnerabilities do not exist.

13.2.11 Memory card attacks could be masked by modifying the zero report AccuBasic script

The machine instructions to perform an Election Zero report are contained on the memory card. An attacker could hide other attacks, such as “pre-stuffing” an election, by always printing zero totals regardless of the current values.

Description: Memory card tampering may be hidden by modifying the zero report AccuBasic script. Before an election, poll workers print an Election Zero report to ensure that all candidates start with zero votes. With

⁹David Wagner, David Jefferson and Matt Bishop, *Security Analysis of the Diebold AccuBasic Interpreter*. Voting Systems Technology Assessment Advisory Board (VSTAAB), February 14, 2006 (URL: <http://www.ss.ca.gov/elections/voting-systems/security-analysis-of-the-diebold-accubasic-interpreter.pdf>).

no memory card authentication (Issue 13.2.1), an attacker could “pre-stuff” a memory card in a way that cannot be detected by looking at memory card vote totals (Issue 13.2.7). By modifying or rewriting the AccuBasic zero report script to print a zero total for all candidates regardless of the value on the memory card, and attacker could circumvent the zero report protection.

Additionally, as discussed by Wagner et al.¹⁰ (Finding 10), this attack can occur no matter the election mode used to transport the memory card. The election mode stored on the card can be modified by an attacker, and the expected boot behavior can be simulated using AccuBasic scripts.

This vulnerability was originally identified by Hursti¹¹ and subsequently confirmed by Wagner et al.¹² and the CA Diebold source code review (5.1.11). Hursti and the CA Diebold source code review provide a step by step explanation of the attack.

Prerequisites: To exploit this vulnerability, the attacker requires the ability to write to the memory card.

Impact: Election results can be pre-stuffed in a way that can only be detected by performing a recount.

Procedural Mitigations: Tamper seals are a potential *partial* mitigation to protect memory cards. For practical limitations see Section 3.5.

Verification: This issue was confirmed through source code analysis and logical reasoning of other confirmed issues.

13.2.12 The AV-OS PC software controls the physical paper ballot box deflector

The ballot box uses a main bin for normal ballots and secondary bin for ballots with write-ins. A compromised AV-OS PC could direct all ballots with write-ins so that they are never counted.

Description: The ballot box deflector controls the storage bin to which a ballot is directed. The deflector motor is controlled by a port on the back of the AV-OS. If the AV-OS PC detects that a ballot contains a write-in, it places the ballot in an alternate bin. If an attacker exploits other vulnerabilities and controls execution, all write-in ballots could be directed to the main bin and thereby will not be counted. Additionally, the alternate bin is used if the election is configured to place specific *overrides* in the alternate bin for review. Just as with ballots with write-ins, an attacker could keep ballots with these overrides from being placed in the alternate bin.

This issue was previously identified in the CA Diebold source code review (5.1.12).

Prerequisites: To exploit this vulnerability, an attacker must have already exploited another vulnerability (e.g., Issue 13.2.10) that allows the machine to be controlled by the attacker.

Impact: All write-in votes will not be counted, and specified overrides will not be brought to election officials’ attention. Additionally, directing all votes to one bin gives an attacker more information in an attack against voter privacy (see Issue 14.4.2).

Procedural Mitigations: Paper ballot recount is a potential *partial* mitigation. For practical limitations see Section 3.5.

Verification: This issue was confirmed through source code analysis and study of hardware manuals.

¹⁰Wagner, Jefferson and Bishop (as in n. 9).

¹¹Hursti, ‘The Black Box Report: Critical Security Issues with Diebold Optical Scan Design’ (as in n. 5).

¹²Wagner, Jefferson and Bishop (as in n. 9).

13.2.13 A voter can cause the AV-OS PC to stop accepting ballots

A voter can perform a “low-tech” attack that renders the AV-OS PC useless until an election official resets the election using the supervisor password.

Description: The CA Diebold red team report indicated that a voter could render an AV-OS PC useless for the remainder of the election day via low-tech attacks. We confirmed a specific low-tech attack that ends an election on the AV-OS PC; however, we were able to resume voting after performing special operations that required access to the supervisor password. However, depending on the availability of knowledgeable staff and election procedures, such an attack could render the device useless for the remainder of the election day. Please refer to the private appendix (24.2.13) for details of the attack.

This issue was previously identified by the CA Diebold red team.

Prerequisites: See private appendix (24.2.13)

Impact: A voter can disrupt an election at a polling place.

Procedural Mitigations: None.

Verification: This issue was confirmed via demonstration with the AV-OS PC and ballot box.

13.2.14 Memory card contents can be accessed from the serial port

When the AV-OS PC is booted into diagnostics mode, an attacker can dump the contents of the memory cards to the serial port on the back of the device. This data includes the obfuscated supervisor password.

Description: The memory card contains the election definition, AccuBasic scripts, and sensitive information such the supervisor password. If an attacker can read the contents of the memory card, a duplicate can be made with malicious modifications (see Issues 13.2.10 and 13.2.11), or the supervisor password for the entire county can be discovered (see Issue 13.2.8)

When the AV-OS PC is booted into diagnostics mode (by holding down the “Yes” and “No” buttons on the front of the device), the user is presented with a number of options, including an option to dump the contents of the memory card to the serial port. An attacker could exploit this vulnerability to retrieve an image of the memory card.

This issue was originally identified by Kiayias et al.¹³ and confirmed by the CA Diebold red team review.

Prerequisites: To exploit this vulnerability, the attacker requires access to the front and back of the AV-OS PC with a memory card containing the election definition.

Impact: With a few minutes of access, an attacker can retrieve the entire memory card image, including the supervisor password used throughout the county. Using a valid memory card image, an attacker can make a duplicate card with malicious modifications.

Procedural Mitigations: The AV-OS PC should remain locked in the ballot box at all times; however, this should only be considered a partial mitigation, as previous studies have reported the locks are easy to pick or otherwise circumvent. Tamper seals do not stop this vulnerability from being exploited, as the attack is purely external to the AV-OS PC.

¹³Kiayias et al., ‘Security Assessment of the Diebold Optical Scan Voting Terminal’ (as in n. 7).

Verification: This issue was confirmed via demonstration with the AV-OS PC. We downloaded the memory card image to a PC; however, we did not extract the supervisor password, as this was performed in previous studies with custom tools.

13.2.15 The AV-OS PC accepts the same ballot multiple times

The ballots used by the AV-OS PC have no identifying marks. An attacker that can maintain hold of a ballot can pull it out of the ballot box and resubmit it.

Description: The AV-OS PC reads identical ballots printed on card stock without unique identifying marks. Previous studies have shown that an attacker can maintain hold on a ballot by attaching a string or a long strip of Post-It notes. This allows an attacker to submit the same ballot multiple times. Because there are no unique identifiers on the ballot, the AV-OS PC will tally the same ballot multiple times.

This issue was originally identified by Kiayias et al.¹⁴.

Prerequisites: To exploit this vulnerability, the attacker requires a few minutes access to the AV-OS PC at the polling place.

Impact: An attacker could vote multiple times with the same ballot. Additionally, an attacker that can make identical copies of ballots could submit multiple votes.

Procedural Mitigations: As a *partial* mitigation, poll workers should pay close attention to voters spending exceedingly long periods of time at the AV-OS PC.

Verification: This issue was confirmed via demonstration with the AV-OS PC.

13.2.16 The AV-OS PC printer can be temporarily disabled without the supervisor password

The AV-OS PC printer is used to print various reports, including the Election Zero report, results summary report, and audit report. An attacker can disable the printer without the supervisor password; however, it will be re-enabled after the AV-OS PC reboots.

Description: The AV-OS PC printer provides important functionality for the poll worker, such as printing the Election Zero report, results summary reports, and audit reports. Disabling the printer may be an important step in mounting an attack, e.g., see step 5 of the 2006 Connecticut study of the AV-OS. This report indicated that the printer could be temporarily disabled by executing supervisor options that require the supervisor password. We confirmed that the printer can be temporarily disabled; however, we also note that the printer can also be disabled by carefully navigating the LCD menus. However, when the machine reboots, the original mapping will be restored.

This issue confirms a finding of Kiayias et al.¹⁵ and extends it in that the supervisor password is not needed to temporarily disable the printer.

Prerequisites: To exploit this issue, the attacker requires access to the front buttons and the ability to power cycle the AV-OS PC. This access may be gained from picking the lock or otherwise circumventing the ballot box design.

¹⁴Kiayias et al., ‘Security Assessment of the Diebold Optical Scan Voting Terminal’ (as in n. 7).

¹⁵Kiayias et al., ‘Security Assessment of the Diebold Optical Scan Voting Terminal’ (as in n. 7).

Impact: The printer can be temporarily disabled. Alone this issue has minimal impact; however it may be used in conjunction with other vulnerabilities to hide traces of an attack.

Procedural Mitigations: The AV-OS PC should be locked in the ballot box at all times. However, this is a *partial* mitigation, as locks can be picked or the ballot box design may be circumvented.

Verification: This issue was confirmed via demonstration with the AV-OS PC and ballot box.

13.3 Premier AV-TSX Vulnerabilities

13.3.1 The AV-TSX will install an unauthenticated bootloader and operating system

The AV-TSX will automatically load a new bootloader or operating system if files with the appropriate name are present on a memory card. Because these files are not authenticated, anyone with access to a memory card can place any software they wish on the AV-TSX.

Description: In order to allow updates to the bootloader and operating system, the AV-TSX scans all inserted memory cards on boot. If the bootloader finds an update to either itself (EBOOT.NB0) or Windows CE (NK.BIN), it erases the previous version of the software and loads the new version from the above file(s). The difficulty, however, is that at no time is the source of these files authenticated. Accordingly, any file with these names placed on the memory card by any party will automatically be loaded and executed by the AV-TSX. This weakness was first noted by Hursti in 2006¹⁶.

This confirms the vulnerability (5.2.1) reported in the California Source Code Report and by Hursti¹⁷.

Prerequisites: An attacker would require approximately 30 seconds and an AV-TSX to execute this attack. Although a lock protects the memory card and power button, these can quickly and easily be bypassed through the use of a tool as simple as a ball-point pen.

Impact: An attacker could gain control of an AV-TSX and execute arbitrary software. Note that because no logging occurs during these operations, attempts to exploit this vulnerability would be known by the election administrators.

Procedural Mitigations: Tamper evident seals are a potential *partial* mitigation to denote access to the memory card slot has occurred. For practical limitations, see Section 3.5.

Verification: This vulnerability was confirmed through source code analysis.

13.3.2 The AV-TSX will install unauthenticated application updates

The AV-TSX will automatically load what it believes to be a new version of `BallotStation.exe` if a file with the correct extension is present on a memory card. Because there is no authentication of this file, an attacker with access to a memory card could run arbitrary software on the AV-TSX.

Description: In addition to providing an update mechanism for the bootloader and the operating system, the AV-TSX allows `BallotStation` software to be updated through a memory card. After booting Windows CE,

¹⁶Harri Hursti, *Diebold TSx Evaluation: Critical Security Issues with Diebold TSx*. Black Box Voting, Unredacted release July 2, 2006, May 11, 2006 (URL: <http://www.bbvdcs.org/reports/BBVreportIIunredacted.pdf>).

¹⁷Hursti, 'Diebold TSx Evaluation: Critical Security Issues with Diebold TSx' (as in n. 16).

the AV-TSX runs `taskman.exe`, which eventually begins `BallotStation`. Before launching `BallotStation`, however, `taskman.exe` scans the contents of the memory card to determine whether or not a new version of `BallotStation` is present. Any file with the extension `.ins` is assumed to be a valid new version of `BallotStation` and is therefore automatically loaded by the voting machine.

As was the problem in the previous vulnerability, there is no checking to determine if an `.ins` file comes from a legitimate (e.g., the county voting headquarters) or illegitimate (e.g., nearly everyone else) source. Accordingly, anyone with 30 seconds of access to a memory card can load whatever software they wish in the place of a legitimate version of `BallotStation.exe`.

This confirms the vulnerability (5.2.2) reported in the California Source Code Report.

Prerequisites: An attacker with 30 seconds of access to any memory card can successfully exploit this vulnerability. The physical attack itself takes no technical skill.

Impact: An attacker could gain control of an AV-TSX and execute arbitrary software. Note that because no logging occurs during these operations, attempts to exploit this vulnerability would not be known by the election administrators.

Procedural Mitigations: Tamper evident seals are a potential *partial* mitigation to denote access to the memory card slot has occurred. For practical limitations, see Section 3.5.

Verification: This vulnerability was confirmed through source code analysis.

13.3.3 There are multiple buffer overflow vulnerabilities in the handling of `.ins` files

The function responsible for loading new versions of `BallotStation` do not operate in a safe manner. Poorly formatted input may cause `BallotStation` to crash. Malicious input may take control of the AV-TSX.

Description: The function used to install new versions of `BallotStation` contains a number of buffer overflow vulnerabilities. As mentioned in the previous Issue (13.3.2), `taskman.exe` searches the memory card for `.ins` files, which it uses to install new versions of the `BallotStation` software. When `taskman.exe` begins reading information from the `.ins` file, it assumes that the information detailing the installation of the code will be not be too large. An adversary could simply change these values (without access to any encryption key) and exploit a buffer overflow vulnerability.

This confirms the vulnerability (5.2.3) reported in the California Source Code Report and Feldman et al¹⁸.

Prerequisites: An attacker would need 30 seconds of time to access a memory card in order to successfully launch this exploit.

Impact: An attacker could gain control of an AV-TSX and execute arbitrary software. This would allow them to control an election and its results, possibly undetectably.

Procedural Mitigations: None.

Verification: This vulnerability was confirmed through source code analysis.

¹⁸ Ariel Feldman, J. Alex Halderman and Edward Felten, 'Security Analysis of the Diebold AccuVote-TS Voting Machine'. In USENIX/ACCURATE Electronic Voting Technology Workshop (EVT). 2007.

13.3.4 An unauthenticated user can read and or tamper with the memory of the AV-TSX

The AV-TSX provides a mechanism that allows anyone to read or write directly from/to memory. This would allow an attacker to easily gain a copy of the compiled source code and/or maliciously change the software running on the AV-TSX.

Description: As originally noted by Hursti¹⁹, the motherboard of the AV-TSX contains a jumper header marked `DEBUG`. When a jumper is installed at this location, a service menu is displayed over the machine's serial port. From this menu, a user can alter settings including a machine's IP address and the boot order. More critically, through the "Mini Monitor" submenu, a user can read/write to the memory of an AV-TSX. An attacker can therefore recover/rewrite the bootloader, operating system, BallotStation and all of the secrets (i.e., cryptographic keys) stored on the machine. Because there is no logging of this operation, an attacker's changes may be untraceable.

Setting this jumper requires physical access to the internals of the AV-TSX. An attacker would need to remove the eight screws holding the plastic casing together. Once the casing was opened, an attacker could complete the circuit at the jumper header by inserting a staple or paper clip. A serial cable would then be attached to the "VIBS KEYPAD" (exposed on the back side of the AV-TSX) and a computer. The amount of time needed to extract information from a machine would vary depending on the data size (seconds for keys, perhaps hours for the operating system).

This confirms the vulnerability (5.2.4) reported in the California Source Code Report.

Prerequisites: An attacker would need unrestricted access to an AV-TSX for 30 seconds to remove the screws and access the casing. The jumper could be set and the cover replaced in an additional 30 seconds. The extraction of data/programs would depend on the size of these items.

Impact: An attacker could gain control of an AV-TSX and execute arbitrary software. This would allow them to control and election and its results, possibly undetectably.

Procedural Mitigations: No election official should be given unsupervised overnight access to equipment such as the AV-TSX (known as a "sleep-over"). Tamper evident seals are a potential *partial* mitigation to denote access to the memory card slot has occurred. For practical limitations, see Section 3.5.

Verification: This vulnerability was confirmed through penetration testing.

13.3.5 The cryptographic keys are not adequately protected

The keys authenticating smart cards and protecting election data are not adequately protected. Accordingly, an attacker can easily break the mechanisms protecting an election.

Description: The AV-TSX uses a number of keys during the process of conducting an election. In order to authenticate itself to a smart card, the AV-TSX stores a *Smart Card Key* (64 bits). When authenticating an inserted smart card, the AV-TSX compares the value of the *Magic Number* (16 bits) stored in its memory against the one presented by the card. To protect the confidentiality of the votes stored on the machine, the AV-TSX uses a third key known as the *Data Key* to encrypt votes. These three keys are defined as `SCKey`, `SCMagic` and `DESKey` in the code. In previous versions of the code, these three keys were hardcoded into the software; however, they can now be set using the Security Card.

These three keys are stored internally in the AV-TSX in the file `bs-security.cf`. Before being saved to

¹⁹Hursti, 'Diebold TSx Evaluation: Critical Security Issues with Diebold TSx' (as in n. 16).

this file, these three values are encrypted using the *System Key* (128 bits) and the well-known AES encryption algorithm in Cipher Block Chaining (CBC) mode. The process of creating the System Key, however, creates a vulnerability.

A function then uses the machine's serial number and the well known MD5 hash algorithm to generate the System Key. Because the machine's serial number is prominently displayed in a number of places (e.g., in the bottom left-hand corner of the touch screen, on the back of the AV-TSX unit, internally in the machine's registry, etc), this value can be observed by any person using the AV-TSX. With this information, the adversary could easily run the MD5 algorithm and generate the System Key.

This confirms the vulnerability (5.2.5) reported in the California Source Code Report.

Prerequisites: Any individual capable of seeing the serial number on a machine could generate its System Key. The theft of the other keys would require access to the file `bs-security.cf`, which could be accomplished by placing a virus on a memory card or through Issue 13.3.4.

Impact: This attack creates the potential for more serious attacks. For instance, malicious software²⁰ (i.e., a virus) could use this knowledge to alter election results, erase system logs and/or leak the keys necessary to create fraudulent smart cards (e.g., Voter Cards).

Procedural Mitigations: Removing the serial number from public display (on the casing and the screen) would lessen an attacker's ability to generate the System Key without touching the voting machine. Because serial numbers are relatively short (6 digits), this is only a marginally effective mitigation. An attacker could calculate all possible system keys in under two seconds before an election occurs.

Verification: This vulnerability was verified through source code analysis.

13.3.6 Malicious software could alter files critical to correctly reporting an election

Because the cryptographic keys are poorly protected, a virus could untraceably change ballot definitions, audit logs and cast ballots.

Description: The correct execution of an election using the AV-TSX relies upon four types of data files:

- **Election Database (.edb) Files:** These files contain ballot definitions. The manual claims that they are named using the digital signature of the file; however, integrity is instead protected by a non-standard message authentication code (MAC) function as first described by Wagner et al²¹. Specifically, an MD5 hash of the file is calculated and then encrypted using AES-128 and the Data Key.
- **Election Resource (.xtr) Files:** These files contain resources including audio and image files, AcuBasic Scripts and other data used in an election. The contents of these files are not encrypted, but are individually authenticated using the MAC construction described above. These files share the same name as their corresponding .edb files, but are identified by their .xtr extension.
- **Ballot Results (.brs) Files:** These files contain the votes cast in on each ballot during the election. The contents of .brs files are encrypted using the AES-128 algorithm in CBC mode using the Data Key.

²⁰Feldman, Halderman and Felten (as in n. 18).

²¹Wagner, Jefferson and Bishop (as in n. 9).

- **Audit Log (.adt) Files:** These files store a record of some of the operations of `BallotStation.exe`. These files are protected in a similar fashion as the Ballot Results, but instead use the System Key during the encryption process.

Given a .brs file, the corresponding .adt, .edb and .xtr files can be determined by examining the fields in the header. The current .brs file is located by reading the contents of the file `assure.ini`, which is also stored on the memory card. If no file is present, Ballot Station “may attempt to load the election corresponding to the ballot box file with the latest timestamp, or simply not load an election at all”²².

Because the key management currently implemented in the AV-TSX is not secure (see Issue 13.3.5), malware running on the AV-TSX could easily alter all of these files without being detected. If such an attack were to happen before an election, the attacker could use this knowledge to insert a vote-changing virus onto the AV-TSX. The attacker may also attempt to alter ballot layout or election resources (e.g., audio tracks) as an attempt to confuse voters. By changing the audit logs, the attacker could then remove any record of their activities.

This confirms the vulnerability (5.2.6) reported in the California Source Code Report.

Prerequisites: A voter or poll worker would need access to an AV-TSX and the System Key. They could then recover the Data Key and exploit this vulnerability.

Impact: An adversary would gain total control of the AV-TSX and an election. If a virus were inserted, malicious control of the election may be achieved.

Procedural Mitigations: None.

Verification: This vulnerability was discovered through source code analysis.

13.3.7 The smart card authentication protocol can easily be broken

The handshake between a smart card and the AV-TSX is not secure. An attacker can easily exploit this weakness as a means of creating their own voter cards.

Description: The authentication protocol between smart cards and the AV-TSX is insecure. The messages exchanged between the voting machine and the smart card can be repeated an unlimited number of times by an attacker to cast multiple votes. The protocol operates as follows:

In an attempt to prevent unauthorized attackers from casting votes on the AV-TSX, the voting machine and smart card exchange messages containing secret values. Upon the insertion of a smart card, the AV-TSX attempts to prove that it is a valid voting machine by presenting two values: the *File ID* (16 bits) and *Smart Card Key* (64 bits). The File ID, which is hardcoded as 0x3d40, is the same for every AV-TSX machine in the country. The Smart Card Key can be reprogrammed using the Security Card. However, in the case that the Smart Card Key is rejected by the Smart Card, the AV-TSX will attempt to use a hardcoded value. This default Smart Card Key, if not changed by an election administrator, is the same for every AV-TSX machine in the country (0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08). Regardless of whether the AV-TSX uses a unique or the default Smart Card Key, an attacker can capture this secret simply by inserting a smart card that logs all messages sent to it. Accordingly, the Smart Card Key is instantly recoverable simply by interacting with the AV-TSX.

²²Diebold Election Systems, *Ballot Station 4.6 Technical Data Package, Revision 1.0*. March 23, 2005.

If the smart card receives the correct File ID and Smart Card Key, it returns a message to the AV-TSX. This message contains information including the type of card inserted (e.g., a Voter Access Card) and the *Magic Number* (16 bits). The AV-TSX checks that the Magic Number stored on the card is the same as the value stored in its memory. If the two values match, the voter is allowed to begin filling out their ballot. If the values do not match, the voter is denied access to the machine. The Magic Number can easily be retrieved by an attacker with access to a smart card reader. Having intercepted the File ID and Smart Card Key above, the attacker can replay these values to the valid smart card. The attacker can then record the magic number and use it to create an unlimited number of forged voter cards.

When a voter casts their ballot, their smart card is rewritten by the AV-TSX to indicate that the card is no longer valid. However, an attacker could program a smart card to pretend to mark itself as invalid. The same card could then be instantly placed back in any AV-TSX to allow the adversary to cast multiple votes.

This confirms the vulnerability (5.2.7) reported in the California Source Code Report.

Prerequisites: This attack could be accomplished by a voter. Because smart card readers/writers are easily portable and included in a number of Personal Digital Assistants (PDAs) and smart phones, such an attack could be easily conducted without notice.

Impact: An attacker could use this weakness to create their own smart cards and cast multiple votes without being detected.

Procedural Mitigations: None.

Verification: The vulnerability was confirmed through source code analysis.

13.3.8 Security Cards can be forged and used to change keys

The smart cards used specifically for security functions can easily be forged by an attacker. Such a card could allow the attacker to make a machine unusable during and votes unreadable after an election.

Description: The AV-TSX uses three keys to protect against unauthorized access: the Data Key (128 bits), the Smart Card Key (64 bits) and the Magic Number (16 bits). The original source code analysis by Kohno et al²³ discovered that all of these values were hardcoded into the software. More recent versions of the software have addressed this vulnerability by allowing election officials to create unique values for each of these keys. By inserting the Security Card into the AV-TSX, an administrator can update these keys.

The key used to authenticate the Security Card is the same default key described in Vulnerability 13.3.7: 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08. While the correct Magic Number is also required to gain access to the AV-TSX, this value is also hardcoded as 0xa535. Accordingly, an adversary can create a Security Card capable of successfully interacting with any AV-TSX unit in the country.

More critically, both the Magic Number and the Smart Card Key used to authenticate the Security Card can not be updated. Accordingly, an administrator who is aware of this vulnerability can do little to prevent it being exploited.

This confirms the vulnerability (5.2.8) reported in the California Source Code Report.

Prerequisites: This attack could be accomplished by a poll worker or county administrator. Because smart card readers/writers are easily portable and included in a number of Personal Digital Assistants (PDAs) and

²³Tadayoshi Kohno et al., 'Analysis of an Electronic Voting System'. In Proceedings of the IEEE Symposium on Security and Privacy. May 2004.

“smart” phones, such an attack could be easily conducted without notice.

Impact: An adversary could affect the election in a number of ways with this attack. By changing the value of the Data Key, the adversary could cause the results of the election to be unreadable by the central tally units. Specifically, because the county officials would not have the correct key to decrypt the results, the votes stored on the machine may not be able to be counted.

Alternatively, the attacker could prevent voters and precinct administrators from being able to use a machine. By changing the Magic Number or the Smart Card key, the attacker could prevent ballots from being cast or voting machines from being used in an election. Moreover, because the Security Cards would typically be kept secure in the county central election facility, it would be unlikely that elections administrators could fix this problem on election day.

Procedural Mitigations: None.

Verification: This vulnerability was confirmed through source code analysis.

13.3.9 The System Setup Menu is accessible without using a smart card

An attacker can access some administrative functions without using the proper smart card. This can allow them to change a number of settings and calibrations, potentially making legitimate voting difficult or impossible.

Description: Access control for the AV-TSX is provided through the use of smart cards (e.g., Voter Access Cards, Security Cards). However, an adversary can gain access to the System Setup Menu without possessing a smart card. Access to the System Setup Menu allows a user to change settings including the time and date, adjust the volume, speed and output device used for audio, change network settings and recalibrate the touch screen.

On boot, the AV-TSX will automatically enter this screen under a number of conditions including hardware failures (e.g., disconnected VVPAT printer, smart card reader failure). We were able to cause an AV-TSX to enter this mode by placing a paper clip into the smart card reader. A similar approach was successfully executed by the red team in the California report.

This confirms the vulnerability (5.2.9) reported in the California Source Code Report.

Prerequisites: This attack could be accomplished by anyone with physical access to an AV-TSX during an election.

Impact: This attack can cause a number of problems in an election. By recalibrating the screen, an adversary can make it difficult or impossible for a voter to fill out a ballot according to their wishes. By changing the audio volume and speed, an adversary can also make access for the disabled challenging or impossible. Network and modem settings can also be changed through this screen, potentially exposing the voting machine to additional vulnerabilities.

Procedural Mitigations: None.

Verification: This vulnerability was confirmed in the source code and by physically disconnecting the printer from the AV-TSX.

13.3.10 The protected counter is not protected

The protected counter acts as a fraud detection mechanism by noting the total number of votes ever cast on an AV-TSX. Because this counter is not protected, an attacker can easily add as many votes as they wish and then hide their attack.

Description: Each AV-TSX stores the total number of votes cast on itself during its entire lifetime in a file. This “protected counter” could be used to detect scenarios including the injection of fraudulent votes by an adversary (over-voting) or machine errors leading to votes not being recorded (under-voting). Unfortunately, the correctness of the counter value in this file can not be trusted.

The protected counter is stored in the unprotected file `\FFX\AccuVote-TS\system.bin`. Because there are no protections provided to this file or its contents, it is in fact not a protected counter. Accordingly, malicious software (e.g., a virus) with access to the file system could easily change the value of this counter without detection.

The method through which this counter is incremented also lacks critical input filters. Even if such a file were protected by access control mechanisms in the operating system, malicious software would be able to increment the counter using negative values (and therefore mask the presence of fraudulent votes).

This confirms the vulnerability (5.2.10) reported in the California Source Code Report, which was originally reported by Feldman et al²⁴. The lack of input validation was not previously noted.

Prerequisites: A poll worker or a voter capable of injecting a virus into the voting machine could successfully launch this attack.

Impact: Exploitation of this vulnerability would make it possible to hide indications that fraudulent votes were entered into the machine.

Procedural Mitigations: None.

Verification: This vulnerability was confirmed through source code analysis.

13.3.11 SSL certificates are not sufficiently protected

SSL is a protocol that can be used to securely communicate between the AV-TSX and the election headquarters. However, the password protecting the cryptographic keys is easily guessable - “diebold”.

Description: While the use of modems to communicate election results is not currently allowed in the State of Ohio, every AV-TSX machine comes equipped with a fully functional modem. Should modem-based reporting ever be allowed, communication between precincts could be cryptographically protected using the Secure Sockets Layer (SSL). The implementation of SSL used in the AV-TSX, OpenSSL, is a well-known and publicly-vetted library.

The secure use of SSL hinges upon the ability to protect the private key stored at both endpoints of communication. These private keys are stored along with the public key in an SSL certificate, which is the file `client.pem` on the AV-TSX. While the private key on the AV-TSX is password protected, the password used to gain access to the key is both weak and hardcoded. Specifically, the manufacturers protect the private key in all AV-TSX machines with the password “diebold”. Such a simple password is easily guessable and therefore provides no real protection to the system.

²⁴Feldman, Halderman and Felten (as in n. 18).

We also suspect that the certificate on all AV-TSX machines is the same. From our analysis of the source code, there does not appear to be a mechanism for certificates to be loaded onto the machine separately or updated. Access to a single AV-TSX, regardless of whether or not modem communications are permitted, may potentially compromise the communications of all AV-TSX units using a modem in the country.

This confirms the vulnerability (5.2.11) reported in the California Source Code Report.

Prerequisites: A voter or poll worker capable of running a malicious program on the AV-TSX could gain access to the private key.

Impact: An adversary can potentially eavesdrop upon and change the results of an election if a modem is used. The GEMS server would also become susceptible to the injection of false election data from computers masquerading as voting machines.

Procedural Mitigations: An AV-TSX can never be attached to a network. This includes overtly by poll workers or covertly by an adversary. Physically removing the modem may provide a straightforward approach to achieving this.

Verification: This vulnerability was discovered through source code analysis.

13.3.12 OpenSSL is not initialized with adequate entropy

SSL can protect communications between an AV-TSX and election headquarters. However, because the “random” key used to encrypt such communication is created in a guessable manner, an attacker can realistically decrypt and change all communications between precincts and election headquarters.

Description: SSL connections can be established between an AV-TSX and the GEMS server to allow secure communication. Before establishing such a connection, the OpenSSL library must be initialized with pseudo-random/unpredictable data in order to ensure the security of the connections. If the pseudo-random/unpredictable data is in fact easily predictable, it becomes possible for an attacker to determine the secrets (i.e., keys) used to protect such communication.

OpenSSL is initialized on boot using two sources. The first takes the contents of the screen. The second records the number of milliseconds since the AV-TSX was booted. Both are easily guessable. In the case of the first, the contents of the screen are identical each time, meaning that this input is deterministic and entirely predictable. The second source is also largely deterministic. Given that a machine will generally boot within a relatively small time window (plus or minus a few seconds), the exact number of clock ticks can quickly be determined.

As noted in the California Report, the GEMS server contains almost identical code for initializing OpenSSL. This code makes a call to `CryptGenRandom()`, a Windows function that is accepted as a good source of pseudo-randomness. This function is also included in the code for the AV-TSX; however, it has been commented out because the version of Windows CE running on these devices does not support its use. Note that Windows CE .NET version 4.2 and beyond support this call.

This confirms the vulnerability (5.2.12) reported in the California Source Code Report.

Prerequisites: An attacker would need access to a phone/network line between the AV-TSX and the GEMS server.

Impact: Because Ohio does not allow for modem communication between the AV-TSX and the GEMS server, the problem would seem solved. Unfortunately, every AV-TSX unit comes equipped with a fully

functioning modem. The adversary may use this device to covertly transmit election results from such a device. Even if SSL is turned on by default, the adversary would be able to intercept such communication and gain access to sensitive information.

Procedural Mitigations: An AV-TSX should never be attached to a network. This includes overtly by poll workers or covertly by an adversary. Physically removing the modem may provide a straightforward approach to achieving this.

Verification: This vulnerability was discovered through source code analysis.

13.3.13 Flaws in the AccuBasic interpreter may allow a virus to control an AV-TSX

The memory cards contains “live code” that is run by a special interpreter on the AV-TSX. This interpreter contains flaws that may compromise the machine’s integrity. As the “live code” comes from the GEMS server, this vulnerability provides potential for viral spread.

Our evaluation has not conclusively confirmed or denied this issue. Due to time limitations and lack of access to specific unredacted reports, we were unable to identify the vulnerabilities found in the previous studies. However, we have no reason to believe the vulnerabilities do not exist.

Description: There are a number of flaws in the way the interpreter handles script contents, e.g., buffer overflows, that allow a script to take over control of the machine. A virus infected GEMS server can create malicious AccuBasic scripts that infect the machine, e.g., it will infect the machine when the supervisor runs the Election Zero report.

This vulnerability was originally discovered by Wagner et al.²⁵ in an extensive study of the AccuBasic interpreter for the California Secretary of State and subsequently confirmed by the CA Diebold source code review (5.2.13). See Wagner et al. for a detailed description.

Prerequisites: An attacker requires the ability to control the AccuBasic scripts written to the memory card. This can be done either by controlling the GEMS server or by gaining write access directly to the memory card; however, the latter may require the attacker to discover the Data Key.

Impact: This vulnerability allows an attacker to escape the restricted control of the AccuBasic interpreter, providing the ability to write directly to vote counters and other critical storage. Most importantly, it allows a virus infected GEMS to extend to the AV-TSX.

Procedural Mitigations: None.

Verification: Our evaluation has not conclusively confirmed or denied this issue. Due to time limitations and lack of access to specific unredacted reports, we were unable to identify the vulnerabilities found in the previous studies. Because these issue has been reported by Wagner et al and then verified in the California report, we have no reason to believe the vulnerabilities do not exist.

13.3.14 Failure to check data on memory cards may allow a virus to run on the AV-TSX

A lack of input checking when loading files from the memory card is potentially dangerous. A virus could take advantage of this weakness to crash or control an AV-TSX.

²⁵Wagner, Jefferson and Bishop (as in n. 9).

Description: When a memory card is inserted into the AV-TSX, the machine searches for configuration information in the file `assure.ini`. This file tells the AV-TSX which election-related files (i.e., `.edb`, `.xtr`) are to be loaded for the current election. Because this configuration information is relatively small in properly formatted files, the BallotStation software assumes that it will be easily stored in a small, fixed-length buffer. Because this file is neither encrypted nor its integrity protected, an adversary could easily replace this configuration information with a large number of characters. By overfilling this buffer, an attacker would be able to cause either the crash of BallotStation or potentially execute arbitrary code on the AV-TSX.

This confirms the vulnerability (5.2.14) reported in the California Source Code Report.

Prerequisites: Anyone with access to a memory card and a laptop could exploit this vulnerability with a few seconds of access.

Impact: An adversary could potentially run arbitrary code on the AV-TSX. This code could change the results of an election.

Procedural Mitigations: None.

Verification: This vulnerability was confirmed through source code analysis and demonstrated through penetration testing.

13.3.15 Unsafe handling of election resource files may allow a virus to control an AV-TSX

Election resource files are trusted to be well-formed. However, weaknesses in the handling of these files can allow an attacker to crash or potentially run malicious code on the AV-TSX.

Description: Election resource files contain additional content to enhance the user's experience during an election. However, due to a lack of input checking on the contents of these files, the AV-TSX is vulnerable to attack. The following two vulnerabilities represent realistic threats to the integrity of the election.

The first weakness is a format string vulnerability. Information used to populate data on the screen, such as the current page number on a multi-page ballot, is stored in language specific RTF files. When the data from these files is read into memory, the system does not handle these strings in a safe manner. For instance, the text `Page %d of %d` in the source code should receive two numbers from an RTF file and use them to replace the `%ds`.²⁶ However, the data destined for this field is loaded via an unsafe call to `sprintf`. Accordingly, replacing the numbers in the RTF file with input such as `“%d%s%s%s%s%s%s%s%s%s”` would cause the AV-TSX to crash. Code could be added to the end of this string to allow the attacker to violate the integrity of the election.

The AV-TSX also uses image files to ease the election process. For instance, when a voter approaches the AV-TSX, the machine displays a graphic of a user inserting their voter card below the text “Insert Card to Begin Voting”. When these bitmap files are loaded into memory, the AV-TSX looks at the header of the image file to determine its size. It then copies the file into a buffer in memory based upon this value. Because the machine trusts the header to accurately represent the real size of the file, there is no real input checking. Accordingly, an attacker can change this value and cause a buffer overflow vulnerability.

This confirms the vulnerability (5.2.15) reported in the California Source Code Report.

Prerequisites: The attacker would need access to the RTF files stored on the GEMS server, the GEMS server

²⁶`%d` is a standard indicator used to tell the system that the data supplied should be treated as a number.

itself or a memory card. As previously mentioned, gaining access to the cryptographic keys protecting the integrity of the memory card's contents is easily achieved (Issue 13.3.5).

Impact: An attacker could exploit either of the vulnerabilities to gain control of an AV-TSX and change the results of an election.

Procedural Mitigations: None.

Verification: These vulnerabilities were confirmed through source code analysis.

13.3.16 Unsafe handling of election database files may allow a virus to control an AV-TSX

Election database files are trusted to be well-formed. However, weaknesses in the handling of these files can allow an attacker to crash or potentially run malicious code on the AV-TSX.

Description: At the end of an election, an AV-TSX can upload the ballots it recorded to the GEMS server. As a confirmation of this upload, the AV-TSX prints a “ticket” containing information stored in the election database (e.g., Date, Time, File Count, etc). Certain weaknesses in the ticket printing routines allow an attacker to potentially take control of the voting machine.

In the first vulnerability, the programmers assumed that certain election information would never exceed 512 bytes in length. Accordingly, the code uses a call to `sprintf` to copy such information into a buffer `buf`. Because there is no input checking on this function, it is possible for more than 512 bytes of data to be copied to `buf`. This buffer overflow would allow an attacker to execute arbitrary code on the AV-TSX.

The second weakness is a format string vulnerability. Because the programmers do not properly handle strings in a safe way, an attacker can inject data such as “`%s%s%s...%s`” and cause BallotStation to crash or execute arbitrary code.

The third vulnerability is also due to unsafe handling of strings. This format string vulnerability uses nearly identical code to the previous vulnerability and therefore can be used to exploit the AV-TSX in the same fashion.

This confirms the vulnerability (5.2.16) reported in the California Source Code Report.

Prerequisites: The attacker would require access to an election database file. Such an attack could occur with access to the GEMS server or a memory card (assuming the attacker was in possession of the correct keys using the weakness noted in Issue 13.3.5).

Impact: An attacker could exploit any one of the above vulnerabilities to change the results of an election.

Procedural Mitigations: None.

Verification: This vulnerability was confirmed through source code analysis.

13.3.17 A voter may be able to gain control of an AV-TSX

A buffer overflow vulnerability in IP address handling may allow a voter to execute arbitrary code on the AV-TSX.

Description: The values for the *IP address*, *Subnet Mask*, *Default Gateway* and *DNS Server* can be set by an administrator through the System Setup Menu. As pointed out in Issue 13.3.9, a voter can also gain access

to these settings without using any smart card. When addresses for the above fields are entered through the touch screen, they are temporarily copied into a buffer. This buffer, which can hold a maximum of 256 characters, is then copied into the system registry such that functions responsible for network communication can access this information.

While some input checking does occur, it is possible for an adversary to cause a buffer overflow using this interface. An IP address, which has the form XXX.XXX.XXX.XXX (where XXX is an integer between 0 and 255), can be written in a manner that passes the format checking while still overflowing the buffer. For instance, an IP address of 192.168.1.1 can legally be represented as 0000 . . . 0192.168.1.1 (where at least 256 zeros precede "192". When such an address is entered, the AV-TSX crashes.

If additional characters (i.e., the full range of alpha-numeric symbols) could be entered, it would be possible to use this vulnerability to execute malicious code. Input validation on these fields would appear to mitigate this additional threat.

This confirms the vulnerability (5.2.17) reported in the California Source Code Report.

Prerequisites: A voter or a poll worker could accomplish this attack by accessing the System Setup Menu (Issue 13.3.9) and then entering a sufficient number of zeros before a valid IP address. Such an attack could easily be accomplished in one minute.

Impact: This attack causes the AV-TSX to crash, necessitating that the device be reset. Continued crashes may cause the device to be unnecessarily removed from service.

Procedural Mitigations: None.

Verification: This vulnerability was first confirmed through source code analysis and then physically confirmed on an active AV-TSX unit.

13.3.18 A malicious GEMS server can cause an AV-TSX to crash on download

The AV-TSX assumes that all data from the GEMS server is well-formed. Poorly formatted data may cause the AV-TSX to crash. Malicious input may allow an attacker to run a virus on the AV-TSX.

Description: One way in which election information can be delivered to a polling place is through direct download from the county's election headquarters. Included in the information transmitted by the GEMS server is a label to be attached to memory cards. This way, these cards can easily be recognized as being ready for use in an election.

As this information is sent to the label printer, the treatment of some of the data fields is unsafe. An attacker with control of the GEMS server (or the network in between the GEMS server and the AV-TSX if the SSL option is not in use) could send `printf` format specifiers (e.g., messages containing "%s") to cause the AV-TSX to crash.

Other format specifiers may allow an adversary to gain control of the AV-TSX.

This confirms the vulnerability (5.2.18) reported in the California Source Code Report.

Prerequisites: An attacker would need to have access to the GEMS server or a link in the network between GEMS and an AV-TSX.

Impact: Because modem/network communication between elections machines and the GEMS server is not permitted, this vulnerability is less likely to be exploited. However, because active modems are placed in all

AV-TSX machines, the possibility of such an attack can not be completely dismissed.

An attacker could use this attack to crash or potentially execute arbitrary software on the AV-TSX. Should the latter be achieved, the integrity of the election would become questionable.

Procedural Mitigations: None.

Verification: This vulnerability was confirmed through source code analysis.

13.3.19 Ballots are stored in the order in which they are cast

When traditional paper ballots are placed in a ballot box, the order in which they were cast is not recreatable. This protects voter privacy. By storing ballots in the order in which they are cast, the AV-TSX potentially allows a specific voter's choices to be identified by an attacker.

Description: When a user casts their ballot, the AV-TSX records votes as records in a `.brs` file. Each result is individually encrypted using AES-128 cipher in CBC mode and the Data Key. Before being encrypted, each ballot is marked with a unique serial number (see Issue 13.3.21), the serial number of the machine on which the ballot was cast and a time stamp. Ballots are then added to the end of the `.brs` file.

The problem with this approach is that a total ordering of the voters using a specific machine is kept. This, in combination with a time stamp identifying when a specific vote was cast, allow an adversary with access to a card the potential to determine how specific ballots were cast. Note that the use of encryption does little to protect the voter in this case due to vulnerabilities such as those discussed in Issue 13.3.5 (e.g., weak, default or poorly protected keys).

This confirms the vulnerability (5.2.19) reported in the California Source Code Report.

Prerequisites: An attacker with access to the AV-TSX who happened to watch when a particular voter cast their ballot could successfully exploit this vulnerability.

Impact: Privacy is critical to the election process. If an attacker can learn the result of a specific ballot, they can threaten individuals (i.e., "voter coercion") to sway election results.

Procedural Mitigations: The machine used by a particular voter must be kept secret from observers.

Verification: This vulnerability was confirmed through source code analysis.

13.3.20 Cast ballots and VVPAT barcodes contain a timestamp

Printed versions of ballots cast on the AV-TSX can contain a timestamp encoded in the barcode. An attacker can therefore determine how a specific voter cast their ballot.

Description: As previously mentioned (Issue 13.3.19), the record of each ballot contains the time at which the ballot was cast. Such information offers an attacker the potential to violate voter privacy by matching a specific ballot to an individual. An adversary with access to the poll book or ExpressPoll and the Data Key (Issue 13.3.5) would be able to match specific ballots to individuals with high probability.

An adversary simply observing the time at which a voter cast their ballot may be able to match this information much more easily. Because the timestamp information is also encoded in a certain type of barcode optionally printed at the bottom of each VVPAT record, an adversary could have an exact matching between individuals and their votes. Whether or not a barcode contains timestamp information depends on the type

of barcode used. If a traditional/1-D (CODE-128) barcode is used, the timestamp is not included. If instead a 2-D (PDF-417) barcode is used, the timestamp information is included.

This confirms the vulnerability (5.2.20) reported in the California Source Code Report.

Prerequisites: An attacker would need to have access to the poll book/ExpressPoll and the Data Key or the VVPAT records with 2-D barcodes to exploit this vulnerability. The easily breakable nature of VVPAT enclosure (Issue 14.8.2) would allow an attacker to accomplish the latter in a matter of seconds and leave little recourse to the polling center.

Impact: Privacy is critical to the election process. If an attacker can learn the result of a specific ballot, they can threaten individuals (i.e., “voter coercion”) to sway election results.

Procedural Mitigations: No procedural fixes are known to fix the problem of individual electronic ballots having timestamps. Only traditional barcodes, which do not include the timestamp, should therefore be used.

Verification: This vulnerability was confirmed through source code analysis.

13.3.21 An attacker can reconstruct the order in which ballots were cast

Ballots stored in the memory of the AV-TSX contain a “random” serial number. However, the process by which this serial number is created is not random. Accordingly, an attacker can use this number to determine the order in which votes were cast.

Description: As a means of preserving voter privacy, each ballot is marked with a serial number generated through a cryptographic pseudo-random function. The resulting serial number on each ballot therefore appears to be random to the casual observer. According to comments in the source code, this function is a 20-bit random permutation generator based on a Feistel network using AES as the round function. The programmer(s) note that this algorithm is described in Bruce Schneier’s book *Applied Cryptography*.

The security of this algorithm and the resulting ballot serial number relies upon the key remaining secret. However, the manner by which this key is generated makes it easily reconstructable by an attacker. The key generation function takes the Data Key and the *BRS Signature* as input. As mentioned in Issue 13.3.6, the signature for a .brs file is created by using the AES-128 algorithm and the Data Key to encrypt the result of the MD5 hash algorithm over the header fields. Unfortunately, the fields in the BRS signature, which include values such as a timestamp, the voting machine’s serial number and the election mode, are all easily predictable. Many of these values are displayed by the voting terminal screen when it is turned on. An adversary capable of recovering the Data Key (see Issue 13.3.5) can therefore easily determine the order in which ballots were cast.

This confirms the vulnerability (5.2.21) reported in the California Source Code Report.

Prerequisites: A voter or poll worker able to recover the Data Key could easily exploit this vulnerability.

Impact: Privacy is critical to the election process. If an attacker can learn the result of a specific ballot, they can threaten individuals (i.e., “voter coercion”) to sway election results.

Procedural Mitigations: None.

Verification: This vulnerability was confirmed through source code analysis.

13.3.22 The AV-TSX does not securely delete files

The AV-TSX does not overwrite the contents of files it deletes. Rather, it deletes them by considering the area they occupy unused. Commodity software can allow an attacker to examine the contents of these supposedly unused areas and recover data.

Description: At certain points during the election process, it may be necessary to delete files that are no longer useful. For example, election officials may chose to delete backup files and audit logs from previous elections to free space. Because election databases contain information that might allow an attacker to link a ballot to a specific voter (Issue 13.3.19), it is necessary to ensure that these files can not be recovered by an adversary.

Throughout the code, the developers use the `DeleteFile()` API call. `DeleteFile()` is a standard Windows call that simply removes the file from a directory listing. However, the contents of the file are not destroyed. An attacker can use publicly available tools to easily recover such data.

This confirms the vulnerability (5.2.22) reported in the California Source Code Report.

Prerequisites: An attacker would need access to a memory card or AV-TSX.

Impact: Privacy is critical to the election process. If an attacker can learn the result of a specific ballot, they can threaten individuals (i.e., “voter coercion”) to sway election results.

Procedural Mitigations: Memory cards should be securely deleted before and after each election.

Verification: This vulnerability was confirmed through source code analysis.

13.3.23 Logic errors in the bootloader create a vulnerability when loading bitmaps

Some of the code contains logical errors. Checks that are intended to protect the software can therefore never do so.

Description: An error in the bounds checking logic when loading images during the boot sequence creates a buffer overflow vulnerability. The code is shown below:²⁷

```
253 void GlibPutPixel(UINT xx, UINT yy, Pixel_t Color)
254 {
255     // Check for library not initialized or (x,y) out of range.
256     if (Framebuffer != FALSE || (xx < USER_X) || (yy < USER_Y))
257     {
258         // Compute the frame buffer offset and write the pixel.
259         FrameBuffer[FB_OFFSET(xx,yy)] = Color;
260     }
261 }
```

On line 256, the program checks that the pixel to be drawn is within the boundaries of the buffer. However, the developer has used the *or* (“||”) operator instead of the *and* (“&&”) operator. Accordingly, the above code does not test whether the data to be written is within the bounds of the buffer. Rather, once confirms that `Framebuffer` exists, it copies the information to that buffer. Bounds checking therefore never occurs.

²⁷The following code was first published verbatim in the public release of the California TTBR.

This confirms the vulnerability (5.2.23) reported in the California Source Code Report.

Prerequisites: An attacker would need to have access to the bootloader to insert a specially crafted bitmap image in order to exploit this vulnerability.

Impact: The impact of this vulnerability is admittedly minimal. If an attacker had access to the bootloader, they could simply load malicious code directly instead of creating a special image file. However, this weakness points to a larger weakness in the software engineering practice. Such an error should have been caught in code review.

Procedural Mitigations: None.

Verification: This vulnerability was verified through source code analysis.

13.3.24 There are a number of errors in the AV-TSX startup code

A number of programming mistakes can be found in the code. For instance, instead of allocating space for a series of characters, the software only allocates space for one. This can potentially be exploited by an attacker.

Description:

```
287 TCHAR name;  
288 _stprintf(&name, _T("\\Storage Card\\%s"), findData.cFileName);
```

The code²⁸ above is an example of a dangerous error found in the AV-TSX. `TCHAR name` represents a single character. However, the call to `_stprintf` on the next line of code writes to `name` as if it were an array or collection of multiple characters. Accordingly, even benign files loaded by the bootloader have the potential to cause the AV-TSX to crash. Such a buffer overflow vulnerability would also potentially allow an adversary to load and execute arbitrary, malicious software on the AV-TSX.

This confirms the vulnerability (5.2.24) reported in the California Source Code Report.

Prerequisites: An attacker would need to have access to the AV-TSX and a PCMCIA memory card.

Impact: An attacker could crash or take control of an AV-TSX by loading malicious software.

Procedural Mitigations: None.

Verification: This vulnerability was confirmed through source code analysis.

²⁸The following code was first published verbatim in the public release of the California TTBR.

PREMIER NEW ISSUES

This chapter describes vulnerabilities we found in the Premier election system, beyond those discovered in the California TTB Source Code Report for Diebold. The public version of this report contains references to sections of the private report. These non-public 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.

14.1 Premier EMP Vulnerabilities

14.1.1 Tampering with a memory card allows attackers to crash or take control of an EMP server.

The EMP server trusts that the information stored in unencrypted/unauthenticated files is benign. Anyone with access to a memory card can create malicious input and use it to crash or gain control over the EMP server.

Description: Originally, multiple AV-TSX units would be set up inside the county election headquarters to accomplish the pre- and post-election tasks of programming and collecting results from memory cards. The *Election Media Processor* (EMP) was designed to expedite these tasks. Consisting of array of up to six PCMCIA card slots, the EMP allows an election administrator to more quickly move ballot definitions from and election results to the GEMS server.

Much of the memory card related source code used in the AV-TSX is also present in the EMP software. So too are many of the vulnerabilities. For instance, Issue 13.3.14, which notes that the PCMCIA card reading function assumes that the contents of `assuri.ini` do not exceed a 1024 character buffer, is present in the EMP software. If an adversary changes the contents of `assure.ini`, the integrity of which are not protected, they can cause a buffer overflow in the EMP. This could be used to potentially change votes on the EMP directly or to use the EMP as a platform to directly attack the GEMS server.

Moreover, this attack could easily be accomplished. An adversary could create their own PCMCIA card (including the identifying AccuVote stickers) and leave it at a polling station. This card would likely make its way back to the EMP where, on insertion, could take control of the election. The infected EMP terminal could then erase the exploit from the memory card to hide its presence from future analysis. Alternatively, an attacker could launch the same attack prior to an election and then infect all memory cards destined for

AV-TSX machines.

The team was able to demonstrate not only that this weakness existed in the source code, but also that it was exploitable. The exploit occurs immediately after a memory card is inserted, leaving no time for an operator to clear the contents of the memory card.

Prerequisites: An attacker would need to have access to any memory card (legitimate or otherwise). This card would need to be read by the EMP.

Impact: The attacker could potentially execute arbitrary code on the EMP. This is more critical than Issue 13.3.14 as the EMP is a central point in the system. Accordingly, the EMP could be used to rapidly distribute a virus.

Moreover, the EMP could be used to infiltrate the GEMS server. Because the two are connected via an Ethernet connection, the EMP could attempt to exploit known vulnerabilities in both the GEMS software and the underlying operating system.

Procedural Mitigations: Chain of custody logs are a potential *partial* mitigation to protect memory cards. For practical limitations, see Section 3.5.

Verification: This vulnerability was confirmed through source code analysis and penetration testing.

14.1.2 A single Data Key is used to encrypt all ballots in a county

The EMP server presents a screen with a single field for the Data Key used to encrypt the results of an election. Because all cards are protected with this key, a single compromised voting machine makes the results on all other memory cards vulnerable to tampering.

Description: As mentioned in Issue 13.3.6, the contents of Ballot Results (.brs) files are encrypted on the AV-TSX using AES-128 in CBC mode with the *Data Key*. The Data Key in each machine should be unique. If such a practice were followed, an attacker able to recover the Data Key from one machine would not help them to attack any other machine.

Previous investigations have noted that it is likely that the same data key is used to encrypt all ballots across a county. An examination of the code on the EMP confirms this suspicion. When the header of the .brs file is read by the EMP, it checks to see whether the hash of the data key in the header matches that one stored in the machine. The only way two such hash values could match would be if the Data Key stored on each unit was the same. If the key hash values do not match, loading fails.

Testing on the EMP server confirms this issue - only one Data Key can be loaded onto the machine.

This vulnerability has not previously been confirmed. California Issue 5.2.5 notes that “all machines within a particular county most likely share the same ... Data Key.”

Prerequisites: An attacker would need to compromise one unit using the Data Key (AV-TSX, EMP).

Impact: An attacker could forge or change results for all precincts in a county with access to a single machine.

Procedural Mitigations: The operator of the EMP should manually enter a unique Data Key for each card that is encoded. This would prevent parallel encoding of cards.

Verification: This vulnerability was confirmed through source code analysis and penetration testing.

14.1.3 The warning that a default key is in use is not sufficient

The EMP server is loaded with a publicly known default Data Key. The warning that this key is in use may not be immediately obvious to all users.

Description: Before an election begins, the EMP user enters the Data Key into a setup menu. The contents of this field (Data Key:) are protected using standard password measures (i.e., each character is represented by a “*” and the contents of the field can not be copied and pasted to another location). As a means of checking to see that the correct key is entered, a second field (Hash:) containing the hash of the Data Key is displayed below.

Should the user enter the default Data Key for encrypting ballots into this field, the background of the Hash field turns pink (RGB: 255, 191, 191). Other key values entered into the Data Key field should turn the background of the Hash field green (RGB: 34, 255, 145). There is no other indication that the default key is being used. The meaning of the background color is explained on page 15 of the EMP 4.6 Users Guide, Revision 2.0 (Section 2.1.16 - Security Tab).

The mechanism is problematic from a number of perspectives. First, one of the members of the team is red/green color-blind. Accordingly, this team member had great difficulty determining whether or not the default key was in use. Secondly, the mechanism that compares the password entered by the EMP user to the default password only checks the first half of the entered password. The size entered for the memcmp, (8 bytes), is half the length of the key.

This mechanism fails to ensure that a large portion of the default key was not used in the Data Key. The first eight bytes can be random and the second eight bytes mirror the default key without turning the background of the Hash field pink. If one would argue that the key from the above case is weak because at least half matches the default value, the mechanism would fail to catch this mirror case (which would be equally as weak).

Moreover, the Hash field only displays the first 4 bytes of the 16 byte MD5 hash. An attacker could quickly generate an equivalent hash value. Because the hash value of the fake key would match that of the expected key, the operator of the EMP server would have no means of knowing the wrong Data Key was being used.

Finally, the concept of a default key is dangerous. At no time should any county rely upon the default key hardcoded into any electronic election system. The default key to the Premier system has long been readily available on the Internet.

Prerequisites: An EMP user not familiar with or able to determine that the background color has a specific security meaning would be susceptible to this weakness. Weak keys could also be generated through this mechanism.

Impact: This weakness would allow weak (or known) Data Keys to be used in an election. If an attacker can find a key with the same hash value for the first 4 bytes, they may cause unreadable memory cards to be distributed to precincts (See Issue 14.1.8).

Procedural Mitigations: The Data Key must be changed upon receipt of the EMP server. Default keys must never be used.

Verification: This vulnerability was confirmed through source code analysis and penetration testing.

14.1.4 A malformed IP address can freeze the EMP server

By entering a malformed IP address, anyone with access to the EMP server may cause the server to become unusable. This problem exists across reboots and reinstallations of the EMP server.

Description: As part of its setup, the EMP software requires that a user enter the IP address or host name of the GEMS server. This allows the EMP server to connect to GEMS when performing its uploading and downloading duties. Should this value need to be changed, the EMP software allows a user to reflect these changes in a field in the Communications Setup menu.

In theory, if the EMP user accidentally enters a malformed IP address or hostname, they should be able to use the Communications Setup menu to correct their error. However, the user may never be given such an opportunity. On startup, the EMP server immediately becomes unresponsive and fails to correctly render the user interface. Because none of the menus have yet been rendered on the screen, the EMP user is never given the opportunity to change this setting back to a correct value.

The value for the GEMS server is stored as an entry in the registry. Because the EMP user is given minimal rights (which is a good security practice), they can not edit the registry to fix this problem. Moreover, unless the administrator understands that the host string is stored in the registry, it is unlikely that they will be able to fix the problem. Because the uninstall program included with the EMP software fails to remove these entries from the registry, this problem persists across reinstallations.

Finally, there is no error checking when a new value is entered into the IP address/host name field. The EMP server always trusts that the user correctly entered the data.

Because of the time limitations of this study, the exact cause of this vulnerability was not located. Please see the Private Appendix 25.1.4 for more details.

Prerequisites: An EMP user, malicious or otherwise, would need to be seated at the terminal.

Impact: The EMP server would be rendered temporarily useless. Because rebooting the machine would have no effect, this attack may make it difficult for memory cards to be delivered to districts on time or for votes to be tallied efficiently.

It is also potentially dangerous that a user can enter unchecked values into the registry. This may be useful as part of another attack.

Procedural Mitigations: None.

Verification: This vulnerability was confirmed through source code analysis and penetration testing.

14.1.5 The contents of the logs on the EMP server are not authenticated

Log files offer officials an audit trail after an election has concluded. However, there is no protection of the log files on the EMP server. These audit trails can therefore easily be forged.

Description: The EMP server keeps logs of many of the operations it performs. For instance, when a blank memory card is inserted and a new ballot definition downloaded, the EMP server creates a log entry. Logging also occurs when cast ballots are uploaded to the GEMS server or when an error (e.g., connection timeout) occurs. These logs provide evidence with which an auditor can reconstruct the events on an election. Similar audit logs exist on the AV-TSX.

Unlike the logs on the AV-TSX, however, the integrity of the EMP logs is not protected. Accordingly, these logs can be altered or forged by an attacker without being detected. These files can also be deleted without notice. The EMP logs therefore have limited forensic value.

Prerequisites: An attacker would need access to the EMP server. Malicious software could easily provide such access. Moreover, if such logs were moved to another machine for analysis, they could be changed by any user there or during transport.

Impact: An attacker can delete evidence of their attack without detection.

Procedural Mitigations: Ensure that all operations performed on the EMP server are observed by multiple parties.

Verification: This vulnerability was confirmed through source code analysis and penetration testing.

14.1.6 The EMP server shares the same default SSL Certificate as the AV-TSX

Communication between the networked devices can be achieved securely if SSL is used properly. However, the same cryptographic keys/certificate stored in every AV-TSX is also stored on the EMP server. An attacker can use this information to communicate with the GEMS server.

Description: The EMP server communicates over a network with the GEMS server. Because the contents of their communication may be sensitive, the link between the two is secured using strong cryptographic mechanisms. OpenSSL provides both encrypted and authenticated communication that matches the recommended best practices for data secrecy and integrity.

As was noted in Issue 13.3.11, it is highly likely that all AV-TSX units in the country contain the same certificate (which has a public/private key pair). Moreover, there does not appear to be any simple mechanism through which a county/voting district can change these certificates. Accordingly, the compromise of a single AV-TSX would allow an attacker to intercept, decrypt and change the contents of communication between any AV-TSX and GEMS server. This represents a misuse of the cryptographic protections.

Like the AV-TSX, the EMP server also stores a certificate allowing secure communication in the file `client.pem`. Unfortunately, the contents of this certificate and the certificate used on all AV-TSXs are exactly the same¹. Accordingly, communications between any AV-TSX OR EMP server and GEMS can be intercepted when a single machine is compromised.

This has particular importance in Ohio, where state law prohibits the use of a modem connection between an AV-TSX and the GEMS server. In spite of this law, an attacker compromising a voting machine can use the information recovered in the attack to launch a second attack on GEMS. Because the GEMS and EMP servers may be run on a semi-public network (e.g., the same network used for desktop machines in the county election headquarters), this weakness may allow a remote attacker to spoof voting results.

As was also reported in Issue 13.3.11, the password “diebold” protecting `client.pem` is weak.

Prerequisites: An attacker would need to compromise any AV-TSX or EMP in order to recover the certificate.

Impact: The attacker could potentially attack the GEMS server by pretending to be an EMP server. The fake EMP server could upload forged or altered voting results, thereby compromising the integrity of the

¹Running `diff`, which compares the contents of two files and reports the differences, returns absolutely no difference between the two files.

election.

Procedural Mitigations: An AV-TSX can never be attached to a network. This includes overtly by poll workers or covertly by an adversary. Physically removing the modem may provide a straightforward approach to achieving this. The certificates in each machine should also be unique.

Verification: This vulnerability was confirmed through source code analysis and penetration testing.

14.1.7 The System Key for the EMP is insecure

The System Key, which protects the other cryptographic keys stored on the EMP, is the same across all machines. More critically, the key is constant.

Description: Like the AV-TSX, the EMP server stores copies of the Data Key in the file `bs-security.cf`. In order to protect these keys, the EMP server encrypts the contents of this file using its System Key. However, the System Key itself is the same on every machine onto which the EMP server is installed. An attacker capable of compromising one EMP server can then easily recover the keys used to protect the AV-TSX from any EMP server.

Like the System Key in the AV-TSX, the System Key for the EMP server is created in a predictable manner. The machine's serial number is fed as input to the MD5 hash algorithm, the deterministic result of which becomes the System Key. On each AV-TSX, this serial number (and therefore the resulting System Key) is unique. However, the serial number used is a fixed value on all machines: 0.

Like the AV-TSX, the EMP server uses a macros to store this value to the registry. However, upon inspection of the registry on machines running the EMP server, the serial number is never actually added. Accordingly, calls to the registry for this value always return NULL, which corresponds to 0. We can not say whether this was intended in the design of the of the EMP or if it was an error. Regardless, this represents a significant threat to security as anyone with access to the file `bs-security.cf` on any machine running the EMP server can subsequently gain access to election results.

Prerequisites: An attacker would need access to the EMP server.

Impact: The attacker knows the System Key. This key could then be used to decrypt `bs-security.cf` to recover the Data Key used in all AV-TSX machines in a county.

Procedural Mitigations: The EMP server must be as protected as the GEMS server.

Verification: This vulnerability was confirmed through source code analysis and penetration testing.

14.1.8 The integrity of the Data Key is not protected

A lack of integrity checking in the EMP server allows an attacker to change the Data Key used to encrypt election results. No alarms are raised when this value is changed until memory cards are placed in an AV-TSX.

Description: The Data Key is encrypted using AES-128 in CBC mode before being stored in `bs-security.cf`. When the EMP server requires the use of these files, it generates its System Key by running the MD5 hash algorithm on its serial number. The contents of `bs-security.cf` are then decrypted and used by the EMP server for their appropriate purposes.

To ensure that the key values correctly decrypted, the EMP server checks the value stored in the 8-byte `id` field. If `id` contains a fixed string, the values assigned with each of the keys are assumed to be correct. However, an attacker can change any part of `bs-security.cf` beyond the first 8-bytes and cause the EMP server to load the incorrect keys.

Assuming that an unknown key were used to encrypt the contents of `bs-security.cf`, an attacker would not be able to change the key to a known value. Rather, a single change in the encrypted contents of the file would lead to unpredictable changes throughout the actual key. For instance, if the attacker changed the contents of previous fields the Data Key would also change. If the attacker instead only changed the encrypted characters corresponding to the Data Key (the last value in the file), only the Data Key would be affected.

The value of this attack would not be in programming a specific key. Rather, by inserting a random key, the attacker would be able to prevent an election from occurring. By encoding all of the memory cards with a random key, none of the AV-TSX units would be able to load the ballot definitions.²

Prerequisites: An attacker would need access to the `bs-security.cf` file on the EMP server.

Impact: An attacker may delay the running of an election as all memory cards encoded with the incorrect data key would be unusable until properly rewritten.

Procedural Mitigations: The hash value displayed on the Security Setup menu should also be monitored constantly (see Issue 14.1.3 for problems with the displayed hash value).

Verification: This vulnerability was confirmed through source code analysis and penetration testing.

14.1.9 GEMS trusts the EMP to deliver correct ballot definitions and results

GEMS delivers ballot definitions and receives election results from the EMP server. However, there is no need for the EMP server to have unencrypted access to either. Because the EMP server is less protected than the GEMS server (no firewall, virus scanner, etc), it becomes a much easier target.

Description: The GEMS server relies upon the EMP server to perform a number of tasks. Before an election occurs, the EMP encodes memory cards with the assorted files necessary to run an election. After the close of an election, the EMP will tally and packages results in a format read by the GEMS server. In order to perform these operations, the EMP server is given access to the Data Key. Accordingly, the EMP can decrypt all of the votes cast on AV-TSX machines.

The major difficulty with this model is that a malicious EMP server could easily switch, forge or drop votes. Because the same Data Key is known by the EMP and all AV-TSX devices and the serial number of the AV-TSX associated with each vote is included in the header of the results file (see Issue 13.3.5), the EMP server can therefore perform all of the operations associated with any AV-TSX and use the correct cryptographic keys to “validate” the results. There would be no means of distinguishing where a vote was written. There is therefore no reason for the GEMS server to believe the accuracy of the results reported from the EMP server.

The option to not use of SSL between the GEMS and EMP servers is additionally troublesome. Even if the chain of custody between all precincts and the EMP server is sufficient to protect the memory cards, an adversary might be able to eavesdrop and change election results if SSL is not enabled. Because the EMP

²When the encrypted Data Key is tampered with, the AV-TSX complains, “Unable to load the election: the active ballot box is encrypted with the wrong media key.”

server removed the encryption protecting the results, the attacker would not need to recover the Data key. Trusting the EMP with the Data Key is therefore an extremely dangerous architectural decision.

Prerequisites: An attacker would need to gain access to the EMP server. This could be achieved by the EMP administrator or remotely by an adversary on the same network using common hacking tools such as Metasploit.

Impact: An attacker would be able to completely control the results of an election.

Procedural Mitigations: SSL must always be used in communication between the EMP and GEMS servers. Additionally, the EMP server should be hardened against compromise to a level at least equal to GEMS.

Verification: This vulnerability was confirmed through source code analysis and penetration testing.

14.1.10 A malicious GEMS server can crash an EMP server

The EMP server trusts that all inputs from the GEMS server are well-formed. Because of a lack of input checking in a number of places, the EMP server is vulnerable to attacks that can crash or gain control of the software.

Description: The EMP server trusts that the election data downloaded from the GEMS server is both well-formed and benign. Accordingly, the EMP becomes vulnerable to attack if the GEMS server or an adversary claiming to be the GEMS server is able to send malformed data. Specifically, the EMP server is vulnerable to a number of format string vulnerabilities. These vulnerabilities occur when the software treats strings of text in an unsafe manner. In this particular example, a malicious administrator can append `printf` format specifiers (e.g., values such as “%s”) to the name of the Voting Center and cause the program to crash.

This particular vulnerability has been confirmed to crash the server on ballot download to the EMP server. Such vulnerabilities may also allow an attacker to take control of the EMP server; however, we did not have time to create such an exploit.

Prerequisites: An attacker would need to gain access to the GEMS server. If the attacker could capture the Data Key (See Issue 13.3.5), they would be able to modify the necessary field directly on the memory card.

Impact: An attacker could crash or potentially take control of the EMP server. In so doing, they may be able to delay or alter the results of an election or spread a virus.

Procedural Mitigations: None.

Verification: This vulnerability was confirmed through source code analysis and penetration testing.

14.1.11 A compromised AV-TSX can crash an EMP server

The EMP server trusts many of the inputs offered by the AV-TSX. A malicious AV-TSX can take advantage of this trust to launch attacks capable of crashing or gaining control over an EMP server.

Description: The EMP server also trusts that the information sent from the AV-TSX is well-formed and benign. The EMP server is therefore vulnerable to attack by an attacker that has compromised an AV-TSX. As was mentioned in Issue 14.1.10, the EMP server is vulnerable to a number of format string vulnerabilities. These vulnerabilities occur when the software treats strings of text in an unsafe manner. In this particular example, a malicious AV-TSX can add `printf` format specifiers (e.g., values such as “%s”) to the name of

the Voting Center and cause the program to crash.

This particular vulnerability has been confirmed to crash the EMP server on when a card containing such a modification is inserted. Such vulnerabilities may also allow an attacker to take control of the EMP server; however, we did not have time to create such an exploit.

Prerequisites: An attacker would need to gain control of a single AV-TSX. If the attacker could capture the Data Key (See Issue 13.3.5), they would be able to modify the necessary field directly on the memory card.

Impact: An attacker could crash or potentially take control of the EMP server. In so doing, they may be able to delay or alter the results of an election or spread a virus.

Procedural Mitigations: None.

Verification: This vulnerability was confirmed through source code analysis and penetration testing.

14.2 Premier General Vulnerabilities

14.2.1 Premier may follow insecure media format procedures

When memory cards are returned to Premier, the “erase” procedure may not entirely remove the contents. This issue may allow an attacker from another state to gain sensitive information about citizens to be used for identity theft.

Description: We received a number of media cards for our security analysis of the Premier system. All cards appeared to be blank; however, upon further investigation, we discovered that memory cards had merely undergone a Windows “Quick Format.” For one of the ExpressPoll compact flash cards, we were able to recover the polling data for a county in another state. We confirmed that a few names and addresses are valid citizens of the county using public online resources. This leads us to believe that memory cards recirculated back to Premier may not be securely formatted. However, this instance may be an isolated occurrence.

Prerequisites: Access to a new memory card from Premier.

Impact: The ExpressPoll data contains potentially private information such as name, address, race, gender, and date of birth. Such information could be used for identify theft.

Procedural Mitigations: All memory cards should be secure erased *before* returning them to Premier.

Verification: This issue was confirmed via demonstration with provided memory cards.

14.3 Premier GEMS Vulnerabilities

14.3.1 Vulnerabilities in Microsoft Windows provided DLL files affect GEMS

GEMS depends on Microsoft Windows libraries (known as DLL files) in various aspects, such as making windows and accessing the election database. If there are vulnerabilities in this COTS software, GEMS is also affected. For example, a recent vulnerability in the Microsoft DLL used to access election databases could allow an attacker to create a an election database that infects GEMS with a virus.

Description: A recent vulnerability of the Jet Data layer targeted towards Microsoft Access³ could also affect GEMS. While publicly available malicious .mdb files are designed to exploit Microsoft Access, a file could just have easily be crafted to exploit GEMS. We have not crafted such a file; however, we have confirmed that the malicious file designed to exploit Access crashes GEMS.

Note that while this vulnerability was made public only after our study began, it is a stark reminder that vulnerabilities are continually discovered in operating systems and supporting libraries. The election software implicitly trusts the operating system and libraries to function correctly.

Prerequisites: Ability to cause GEMS to open the malicious .mdb file.

Impact: For the case of the Microsoft Jet vulnerability, anyone who has access to modify a .mdb file can control an election.

Procedural Mitigations: None.

Verification: This issue was confirmed via demonstration with the GEMS server.

14.3.2 Many GEMS servers state-wide use the same BIOS password

Premier maintains a support contract with many of the Ohio counties using Premier equipment. Most counties use the same BIOS password. Knowledge of the BIOS password allows an attacker to boot from external media to subvert the system.

Description: Premier maintains a support contract with many of the Ohio counties using Premier equipment. 40 of the 48 GEMS servers deployed have the same BIOS password. An attacker discovering the BIOS password in one county has a high probability of gaining access BIOS access to the GEMS server in another county. Knowledge of the BIOS password enables an attacker to boot from external media to subvert the system (see Issue 14.7.3).

Prerequisites: Access to the GEMS BIOS password in one county.

Impact: The GEMS server in another county can be compromised.

Procedural Mitigations: Each GEMS server should have a different BIOS password.

Verification: This issue was confirmed through personal correspondence with Ohio Secretary Of State Office employees.⁴.

14.4 Premier AV-OS PC Vulnerabilities

14.4.1 Forged ballots can be forced into the ballot box

When the AV-OS PC scanner detects jammed paper, the roller motors loosen their grasp and the ballot can be pushed into the ballot box. While the vote is not recorded, an attacker can fill the ballot box with forged ballots that may be accepted as official upon recount.

³Microsoft JET – Remote hacker manual control. (URL: <http://www.beskering.com/security/2007/11/17/73>).

⁴Personal correspondence with Ohio Secretary of State Office employee. December 4, 2007.

Description: The AV-OS PC scanner senses when a ballot is not feeding properly through the reader. In order to avoid motor burnout and allow a paper jam to be fixed, the rollers disengage, allowing paper to be unjammed. An attacker can cause the AV-OS to detect a jammed ballot by simply firmly holding on to a ballot. Once the rollers disengage, the ballot can be manually fed through the scanner. As all ballots are printed on card stock, the another ballot is strong enough to push the illegitimate ballot into the ballot box. We were unable to find any reference to the jammed scanner in the audit report.

Prerequisites: An attacker requires access to the AV-OS PC and ballot box while it locked by physical key. Note that the motors are disengaged while the AV-OS PC is powered off. Finally, the ballot box contains a lockable panel to seal the ballot box when the AV-OS PC is not in place. If this panel is in place, this vulnerability cannot be used to force ballots into the ballot box.

Impact: An unsupervised attacker with a few seconds access to a locked AV-OS PC and ballot box can feed forged ballots into the ballot box which may be accepted as official upon recount. Note that ballots must be fed in one at a time, therefore the required time is directly related to the number of ballots forced into the ballot box.

Procedural Mitigations: The ballot feed panel should always be locked when the ballot box is not supervised. Note that simply locking the AV-OS PC in place is not sufficient. This is a *partial* mitigation, as locks may be picked.

Verification: This issue was confirmed via demonstration with the AV-OS PC and ballot box.

14.4.2 The AV-OS PC ballot box collecting votes allows vote order to be reconstructed

The AV-OS PC ballot box sorts ballots based on the existence of a write-in candidate; one bin is kept for each. Within these bins, all ballots lay in the order they were cast. Thus, specific voters may be targeted by noting the order in which they vote and examining the pile of ballots at close of election.

Description: During election day, the AV-OS PC is placed in a special ballot box. The box contains a number of physically locked compartments to keep cast ballots sealed inside. This box also contains a data port that allows the AV-OS PC to control a motor and plastic flap that sorts ballots into one of two bins depending on the existence of a write-in candidate.

When a ballot is cast, it falls into the designated bin. We examined the ballot box and discovered that there are no mechanisms within to inhibit or otherwise change the order in which ballots fall. As a result, at the close of an election, many ballots will be cast at the bottom of the non-write-in bin in the same order they were cast.

We numbered and cast 10 ballots through the AV-OS PC. Upon checking the ballot box, we examined the pile and found that the order was maintained. A watchful adversary who had access to the ballots after they had been cast would be able to report, with non-negligible 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 AV-OS PC, 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 the voter had voted. The probability of identifying a specific voter increases as the number of ballots containing write-in votes decreases.

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.



Figure 14.1: The same key works in both the Hart (left) and Premier (right) ballot boxes.

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

Verification: This issue was confirmed via demonstration with the AV-OS PC and ballot box.

14.4.3 The Hart ballot box key works in the Premier ballot box

While the Premier and Hart ballot boxes are significantly different, the keys we received work in both. If what we observed represents a larger trend, an attacker gaining access to the ballot box key in one county can compromise other ballot boxes within the state, regardless of the vendor.

Description: Both Premier and Hart InterCivic have precinct optical scanners using large ballot boxes on top of which a small scanner device is secured. The ballot box appears to be custom made for each vendor and, in the case of Premier, performs special functionality (sorting write-in ballots from non-write-in ballots).

We received keys for the ballot boxes from the Secretary of State and discovered that they are interchangeable between the two boxes. Furthermore, each ballot box has multiple locks, all of which are accessed by the same key. Figure 14.1 shows the same key working in both ballot boxes. If what we observed represents a larger trend, an attacker gaining access to the ballot box key in one county can compromise other ballot boxes within the state, regardless of the vendor.

Prerequisites: If the ballot boxes use the same key, an attacker requires access to the ballot box key in just one county.

Impact: Ballot boxes in other counties within the state, regardless of the vendor, may be compromisable.

Procedural Mitigations: Each ballot box should have unique keys.

Verification: This issue was confirmed via demonstration with the ballot box.

14.5 Premier VCEncoder Vulnerabilities

14.5.1 Access to the Voter Card Encoder is not protected by a PIN

With access to a Supervisor Card, a Voter Card Encoder can be enable to create legitimate Voter Smart Cards. Similar actions on the AV-TSX require a PIN. Such an operation should be additionally protected with a PIN to provide protection against stolen cards.

Description: The Voter Card Encoder allows poll workers to create valid Voter Smart Cards during an election. Protection for this device is provided in the form of the Supervisor Smart Card. This card, to which access should be limited, also allows administrative operations to be conducted on AV-TSX units. Furthermore, all supervisor actions performed on the AV-TSX require a PIN.

Even the best procedures may occasionally fail to protect the Supervisor Card. In such an event, there is no protection against an attacker activating a Voter Card Encoder and creating an unlimited number of legitimate voter cards.

Prerequisites: An attacker would need to gain access to a Supervisor Smart Card and a Voter Card Encoder.

Impact: The attacker would be able to create an unlimited number of legitimate voter cards.

Procedural Mitigations: Chain of custody logs are a potential *partial* mitigation to protect Voter Card Encoders. For practical limitations, see Section 3.5.

Verification: This vulnerability was confirmed through penetration testing.

14.5.2 Once the VCE is enabled, no mechanism prevents smart cards from being encoded

Once a Voter Card Encoder is enabled, nothing other than possession prevents it from being used to create legitimate Voter Smart Cards.

Description: When the Voter Card Encoder is enabled using the Supervisor Smart Card, poll workers can use this device to create legitimate Voter Smart Cards. However, after this initial access check, any individual with access to the Voter Card Encoder can create voter cards.

If a poll worker were to place the Voter Card Encoder on a desk, an attacker may “accidentally” drop the contents of their bag on the desk. Without arousing much suspicion, the attacker could then take the Voter Card Encoder into their possession. After leaving the polling place, the attacker could encode legitimate Voter Smart Cards from the parking lot and provide them to colluding adversaries in order to cast extra votes.

Such an attack need not happen on election day. We were able to activate an unmodified Voter Card Encoder, turn the power on and off and were still able to create legitimate voter cards over three weeks later.

Prerequisites: An attacker would need to take an active Voter Card Encoder before or during an election.

Impact: The attacker could create an unlimited number of legitimate Voter Smart Cards.

Procedural Mitigations: The Voter Card Encoder should not be out of the physical control of trusted elections officials/poll workers at any time. Chain of custody logs are a potential *partial* mitigation to protect these devices. For practical limitations, see Section 3.5.



Figure 14.2: A Voter Card Encoder running arbitrary software.

Verification: This vulnerability was confirmed through penetration testing.

14.5.3 Software can be loaded by anyone with access to the VCE

Installing new software on a Voter Card Encoder lacks any authentication mechanisms. Simply by pressing the “Off” button, followed by “Yes” will replace the software stored on the device.

Description: Like the other software components of any system, the Voter Card Encoder may require software updates over the course of its lifetime. In order to facilitate such an operation, the Voter Card Encoder can receive software updates using a 9-pin serial cable attached to a PC. To load new software onto the Voter Card Encoder, a user simply turns the device off. When the user presses the off button again, the Voter Card Encoder asks the user to press “Yes” if they would like new software to be loaded.

The problem with this mechanism is that it lacks any authentication of the new software loaded onto the Voter Card Encoder. As a demonstration of this issue, we created and loaded new software. Where the standard software would require a user to activate the Voter Card Encoder with a Supervisor Smart Card, we allowed any card (even unrecognizable formats) to enable the device. An adversary could therefore steal a Voter Card Encoder, load their own software and then create valid Voter Smart Cards. Figure 14.2 shows an example of such software.

Alternatively, we could have used this vulnerability to encode any type of card we wished. For instance, an attacker could easily create Central Administrator, Security or Supervisor Cards by modifying the software running on the Voter Card Encoder.

Prerequisites: An attacker would need access to a Voter Card Encoder.

Impact: An attacker could create valid Voter Smart Cards without the Voter Card Encoder being enabled by the Supervisor Smart Card.

Procedural Mitigations: None.

Verification: This vulnerability was confirmed through penetration testing.

14.5.4 The VCE accepts the default smart card key

When attempting to authenticate an inserted smart Card, the Voter Card Encoder first attempts to see if the card contains the publicly known default password.

Description: The Voter Card Encoder authenticates inserted smart cards as a means of access control. If the inserted smart card does not contain the proper password, the card is rejected. However, like the AV-TSX, the Voter Card Encoder accepts smart cards with the hardcoded default password (0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08). Moreover, the Voter Card Encoder checks for this password first, only attempting to see if the password set by a previous Security Smart Card is present after the default check fails.

Because of the obviousness and public nature of this password, an attacker can easily forge the smart cards needed to gain access to the Voter Card Encoder.

Prerequisites: The attacker needs access to the Voter Card Encoder and the ability to write their own smart cards.

Impact: An attacker can gain complete administrative access of the Voter Card Controller and use it to create legitimate (to cast fraudulent ballots) and illegitimate (to delay or prevent an election) smart cards.

Procedural Mitigations: Chain of custody logs are a potential *partial* mitigation to protect Voter Card Encoders. For practical limitations, see Section 3.5.

Verification: This vulnerability was confirmed through penetration testing.

14.6 Premier ExpressPoll Vulnerabilities

14.6.1 The ExpressPoll runs a webserver

If an attacker is able to gain control of the ExpressPoll, the existing webserver can be modified to create an undetectable back door for future accesses. We were unable to determine why the ExpressPoll is running a webserver.

Description: The ExpressPoll device replaces paper poll books at the polling place. Because precincts may have multiple ExpressPoll units, each device has a Ethernet NIC, which allows devices to synchronize poll book data, e.g., who has voted. We discovered that the ExpressPoll is running a webserver on port 80 (port 443 was also open; however, it did not respond to a web browser). The server displays a default webpage for the Microsoft Platform Builder. We were unable to determine why the ExpressPoll is running the webserver.

The existence of the webserver poses a significant threat to the Express Poll. Once an attacker gains access to the ExpressPoll, possibly exploiting other vulnerabilities mentioned in this report, the webserver can be patched to provide a back door. More importantly, the attacker can revert attacks that visibly change the system, e.g., Issues 14.6.3 and 14.6.4, and maintain access under the guise of the seemingly legitimate webserver.

Prerequisites: An attacker requires the ability to connect to the Ethernet port on the bottom of the ExpressPoll.

Impact: The existence of the legitimate webserver may hide the existence of a persistent back door created after exploiting other vulnerabilities.

Procedural Mitigations: None.

Verification: This vulnerability was discovered through experimentation with the ExpressPoll.

14.6.2 The ExpressPoll will install an unauthenticated bootloader and operating system

The ExpressPoll will automatically load a new bootloader or operating system if files with the appropriate name are present on a memory card. Because these files are not authenticated, anyone with access to a memory card can place any software they wish on the ExpressPoll.

Description: In order to allow updates to the bootloader and operating system, the ExpressPoll scans all inserted memory cards (both PCMCIA and CF) on boot. If the bootloader finds an update to either itself (EBOOT.BIN) or Windows CE (NK.BIN), it erases the previous version of the software and loads the new version from the above file(s). The difficulty, however, is that at no time is the source of these files authenticated. Accordingly, any file with these names placed on the memory card by any party will automatically be loaded and executed by the ExpressPoll. This weakness is similar to Issue 13.3.1 on the AV-TSX.

Finally, there is a chance that placing the Windows CE file (NK.BIN) will not replace the current operating system, but rather only boot from it. Due to the potentially destructive nature of the testing, we verified that the files are accepted, but did not allow the process to be completed. However, either functionalities, booting as runtime image or direct flashing, offer the same potential. Once booted, the runtime image can flash itself to permanent memory.

Prerequisites: An attacker would require approximately 30 seconds and an ExpressPoll to execute this attack.

Impact: An attacker could gain control of an ExpressPoll and execute arbitrary software.

Procedural Mitigations: Tamper evident seals are a potential *partial* mitigation to denote access to the memory card slot has occurred. For practical limitations, see Section 3.5.

Verification: This issue was confirmed via demonstration with the ExpressPoll.

14.6.3 The ExpressPoll uses an unprotected database for poll data

The ExpressPoll database contains information about all registered voters in an area and creates Voter Access Cards for use in the AV-TSX on election day. An attacker can modify the database to include additional names to allow extra voters in a precinct and/or remove names to prevent voters from voting.

Description: The ExpressPoll uses a single DB3 database file to store all the list of all legitimate voters and their designated precincts. The database file is not limited to just one precinct and can span large regions. The file has no protection to prevent malicious modification. An attacker with access to the database file can append new voter records. On election day, these records will allow illegitimate voters to vote in a precinct. An attacker can also remove voter records, thereby disenfranchising voters from their right to vote.

The database file also includes user and supervisor passwords to access various administrative options within the software. Therefore, an attacker with access to the poll database can obtain or change the user and supervisor passwords. The attacker can even disable the need for a supervisor smartcard. These changes allow an attacker to use the ExpressPoll to manipulate poll data.

Prerequisites: An attacker requires 30 seconds of access to the ExpressPoll memory card.

Impact: An attacker has complete control over who can and cannot vote in an election. An attacker can add voters to a precinct and/or prevent specific voters from voting.

Procedural Mitigations: Tamper evident seals are a potential *partial* mitigation to denote access to the memory card slot has occurred. For practical limitations, see Section 3.5.

Verification: This issue was confirmed via demonstration with the ExpressPoll.

14.6.4 The ExpressPoll uses an unprotected resource file

The ExpressPoll interface is defined by a single resource file contained on the memory card. An attacker with access to the memory card can modify this file to extend the functionality of the ExpressPoll application and install malicious software.

Description: The ExpressPoll software uses a .NET resource file to describe the layout and functionality of options in the user interface. The file's authenticity is not verified, and an attacker with access to the file can manipulate the feature functionality of the user interface. Buttons can be rebound to different functionality, and new buttons can be created. For example, a button can be added to launch `explorer.exe`, thereby giving the attacker full access to the system.

Prerequisites: An attacker requires 30 seconds of access to the ExpressPoll and its memory card. Depending on the attack, access to the memory card may be sufficient.

Impact: The attacker can install malicious software on the ExpressPoll and/or make arbitrary modifications to any software on the system.

Procedural Mitigations: Tamper evident seals are a potential *partial* mitigation to denote access to the memory card slot has occurred. For practical limitations, see Section 3.5.

Verification: This issue was confirmed via demonstration with the ExpressPoll.

14.6.5 The ExpressPoll can be rendered useless in transit

The ExpressPoll can be charged while locked in the carrying case (for the case we were provided). An attacker can connect the case to an incorrect power supply and possibly physically destroy the device.

Description: The ExpressPoll carrying case we were provided contains an external power socket presumably used to keep the batteries charged while in storage. If the power cable connecting the external socket to the ExpressPoll is connected during transit, an attacker can destroy the device by connecting an incorrect power source. Tamper evident seals and locks may prevent election officials from testing the device before election day. Note, the ExpressPoll case may vary by precinct, therefore this attack may not generally apply.

Prerequisites: An attacker must have access to the ExpressPoll while locked in a tamper evident sealed case with the power cable connected.

Impact: Poll workers will not be able to verify voters and make Voter Access Cards until a replacement ExpressPoll arrives.

Procedural Mitigations: Test the ExpressPoll and disconnect the power cable before it leaves the county office.

Verification: This issue was confirmed via demonstration with the ExpressPoll; however, we did not attempt to connect an incorrect power supply, as it would destroy the device.

14.6.6 The ExpressPoll audit logs are not protected

The ExpressPoll audit log is contained in two unprotected files on the memory card. An attacker can modify these files to hide malicious activities.

Description: The ExpressPoll is used to authenticate voters on election day. It logs all activities, including login attempts and modification of voter information, to an unprotected DB3 database file. System exceptions are stored in a separate .xml text file, also unprotected. Additionally, if either file is deleted, the ExpressPoll creates a new file without indicating an error to the user.

An attacker with access to the log files could remove all traces of malicious activity. For example, if a malicious poll worker changes a voter's status back to "unvoted," the corresponding log entry can be removed or changed to an otherwise benign event.

Prerequisites: An attacker requires 30 seconds of access to the memory card.

Impact: An attacker could modify or delete log entries to hide malicious activity.

Procedural Mitigations: Tamper evident seals are a potential *partial* mitigation to denote access to the memory card slot has occurred. For practical limitations, see Section 3.5.

Verification: This issue was confirmed via demonstration with the ExpressPoll.

14.6.7 ExpressPoll audit logs violate voter privacy

The ExpressPoll audit log records a voter identifier when a voter is given a Voter Access Card. This identifier corresponds to a field in the voter information database and can be used to determine the order and time in which voters enter the polling place.

Description: The ExpressPoll is used to authenticate voters as they arrive at the polling place. Once authenticated, the voter is given a Voter Access card, and the voter information database (discussed in Issue 14.6.3) is updated to indicate the voter has already entered the polling place. Along with this update, the ExpressPoll appends the activity audit log (discussed in Issue 14.6.6) indicating the `voterId`. The `voterId` field recorded in the audit log matches the field in the voter information database. As the audit log is appended, the order voters enter the polling place is captured with a sequence number, and while a timestamp is not recorded for these entries, other entries, e.g., power-on, include a timestamp. Hence, an attacker can derive approximate times for voter entries.

The memory card we received with the provided ExpressPoll was blank. However, due to the erase method used on the memory card (see Issue 14.2.1), we were able to retrieve a legitimate voter information database. Using this we were able to analyze the audit log and correlate the `voterID`.

Prerequisites: An attacker requires access to the voter information database and the audit log database. For additional time information, the attacker requires the ability to power cycle the ExpressPoll multiple times during the election day.

Impact: An attacker can determine the order in which voters enter the polling place, possibly an approximate time.

Procedural Mitigations: Chain of custody logs are a potential *partial* mitigation. For practical limitations, see Section 3.5.

Verification: This issue was confirmed via demonstration with the ExpressPoll.

14.7 Premier Digital Guardian Vulnerabilities

14.7.1 Users with administrative access can circumvent boot restrictions

If an attacker can boot from a different operating system, e.g., a “live CD,” the system can be modified without Digital Guardian protections. Therefore, Digital Guardian contains explicit policy to keep the Windows boot time selection from modification. Due to a vulnerability in the Digital Guardian policy, a user with administrative access can circumvent protections to modify the boot time selection and load an operating system from CD.

Description: Digital Guardian cannot provide protection to the file system if it is accessed by another operating system. For example, if an attacker boots the system off of a “live CD,” files can be modified without restriction. The Digital Guardian policy explicitly denies all users access to the `Windows boot.ini` file. This file is used by Windows to select from multiple operating systems that may be installed on the machine. It is also commonly used to allow the user to boot to the “Recovery Console” without booting from the Windows installation CDROM.

While the policy denies access to `boot.ini`, it does not deny access to `ntldr`. The `ntldr` file is a executable used by Windows to bootstrap loading the core operating system, e.g., the Windows kernel. `ntldr` has a hardcoded string to look for `boot.ini`. The `boot.ini` determines which operating system to load. An attacker can use a file (hex) editor to modify `ntldr` to look for a different file, e.g., `b00t.ini`, that is controlled by the attacker.

Controlling the boot entries provides an attacker great flexibility. If the recovery console software is installed, an entry can be created that allows a user with the administrator password to disable Digital Guardian. However, even more powerful, the adversary can use well known open source bootloaders that run from the `boot.ini` file (e.g., Grub4DOS) to boot from a CDROM. This is particularly useful when the system BIOS is configured to not allow these boot options. Once the attacker boots to this “portable operating system,” he has complete control over the Windows file system. Possible attacks include modifying files (e.g., the election database), retrieving system passwords, and disabling Digital Guardian. See Issue 14.7.3 for more information about disabling Digital Guardian from removable boot media.

Prerequisites: The attacker requires a few minutes time and access to a user account with administrative privileges. Additionally, the attacker requires a way to copy necessary files and tools to the GEMS server, e.g., CDROM, USB drive, network. The CDROM drive is required if the attacker wishes to boot from a CDROM.

Impact: The attacker can boot a “portable operating system” to gain complete control over the system. Possible attacks include modifying files (e.g., the election database), retrieving system passwords, and disabling Digital Guardian. Note that *GEMSUser* has administrative access and can perform this attack (see Issue 14.7.2).

Procedural Mitigations: None.

Verification: This issue was confirmed via demonstration with the GEMS server.

14.7.2 The *GEMSUser* account is in the *Administrators* group

The *GEMSUser* is configured as a system administrator to the GEMS server. This allows the user to exploit other vulnerabilities, possibly disabling Digital Guardian.

Description: Along with the Digital Guardian policy, three Windows system users were created: *Administrator*, *GEMSadmin*, and *GEMSUser*. The *GEMSUser* account is a member of the system *Administrators* group. This allows the user to perform any administrative action on the machine. While Digital Guardian protects the system from some commands, this configuration allows normal users to exploit vulnerabilities they would otherwise be unable to perform. For example, due to this vulnerability, *GEMSUser* can completely subvert the system using Issue 14.7.1.

Prerequisites: Access to the *GEMSUser* password and the ability to log on to the GEMS server either via physical access or network connection.

Impact: *GEMSUser* can perform any administrative action on the system, possibly disabling Digital Guardian by exploiting other vulnerabilities (e.g., Issue 14.7.1).

Procedural Mitigations: Remove *GEMSUser* from the *Administrators* group.

Verification: This issue was confirmed via demonstration with the GEMS server.

14.7.3 Digital Guardian can be disabled by booting from external media

An attacker with the ability to boot from external media, e.g., a CDROM, can access the GEMS file system to disable and later re-enable Digital Guardian.

Description: If an attacker can boot from external media, e.g., a CDROM drive, Digital Guardian can be disabled. Booting to this media can be done either via BIOS boot menus or the operating system loader (see Issue 14.7.1).

Various “portable operating systems” (also known as “live CDs”) can be used to disable Digital Guardian. If the *Administrator* account password is known, the attacker can boot from a Windows installation CD and select the recovery option. If the *Administrator* account password is not known, the attacker can boot to one of many well known password recovery/reset utility CDs to obtain/reset the *Administrator* password. The recovery console prompts for the administrator’s password; however, once entered, the attacker can disable the Digital Guardian device drivers. When the system reboots, Digital Guardian is disabled. The same process can be used to re-enable Digital Guardian after an attack occurs.

If the attacker prefers not to use the *Administrator* account password, one of many Linux “live CDs” with a command line Windows registry editor can be used to keep the Digital Guardian drivers from loading. Note that while Windows is running, Windows protections do not allow modification of the part of the registry where driver settings reside. While this option requires slightly more sophistication, it does not require a password.

Finally, there exist ways to install Linux from within Windows. Therefore, it may be possible to mount this attack without booting from external media; however, due to time restrictions, we have not confirmed this.

Prerequisites: The attacker requires either BIOS password or Issue 14.7.1 and access to external media device, e.g. CDROM.

Impact: Digital Guardian will be disabled and can be re-enabled after an attack.

Procedural Mitigations: None.

Verification: This issue was confirmed via demonstration with the GEMS server.

14.7.4 An administrative user can disable the Digital Guardian device drivers once

The Digital Guardian policy is enforced by a set of system device drivers. We found that the device drivers can be disabled once after Digital Guardian is installed; however, after re-enabling the drivers, subsequent attempts to disable the drivers fail. However, the ability to disable Digital Guardian even once could allow an attacker to replace Digital Guardian with a malicious version.

Description: Digital Guardian implements its reference monitor using four device drivers that hook into various hardware and software interrupts in the OS kernel. The drivers are “hidden,” but can still be viewed in “Device Manager.” An attacker can easily identify the specific drivers that relate to Digital Guardian by looking at the “Provider” string for all drivers. Once identified, the attacker uses GUI controls to disable the drivers.

Through experimentation, we found that these drivers can be disabled only one time. After the drivers are re-enabled, attempts to disable the drivers fail. However, we were able to repeat the attack by restoring the hard drive disk image to the systems original state. Without access to the Digital Guardian source code we cannot understand why this occurs. We additionally tried removing and reinstalling the agent using the Digital Guardian console; however, in our tests this did not allow us to disable the drivers a second time.

In many cases, an attacker only needs to disable Digital Guardian one time. Once it is disabled, the attacker can replace the drivers with ones containing a backdoor. Note that any user with administrative access can perform this attack; this includes *GEMSUser* (see Issue 14.7.2). Finally, we believe this problem cannot be fixed by changing the Digital Guardian policy and should be considered a flaw in the Digital Guardian software.

Prerequisites: An attacker requires access to a user account with administrative access on a GEMS server where Digital Guardian has not already been disabled via this issue.

Impact: Users with administrative access, including *GEMSUser* can easily disable Digital Guardian.

Procedural Mitigations: None.

Verification: This issue was confirmed via demonstration with the GEMS server.

14.7.5 Users with administrative access can circumvent many Digital Guardian controls

Digital Guardian is used to restrict even users with administrative access. However, within Windows, there exists a user with even more privileges than *Administrator*. While no one can log in as this user, a user with administrative access can execute commands to gain these special privileges. Many of the Digital Guardian policy rules do not restrict a user with these privileges.

Description: The Digital Guardian policy explicitly controls what the *Administrator* user can do on the system. However, within Windows there exists a user with even more privileges than *Administrator*: *SYSTEM*. Many system services run as *SYSTEM*. While a user cannot log in as *SYSTEM*, there are well known ways for a user with administrative access to open a command line as *SYSTEM*. We used the command line task scheduler to gain such a shell. We stress that while simply keeping users from executing this task sched-

ule stops this specific method of gaining *SYSTEM* access, it does not solve the problem. See Issue 14.7.6 for further discussion on the drawbacks of protecting a system by denying execution of a specific set of applications.

With the current system configuration, an attacker logged in as either *Administrator* or *GEMSUser* can easily create a new *SYSTEM* shell. Once the attacker gains this access, many Digital Guardian restrictions can be circumvented. We were able to open the main system management console, which is explicitly denied by the policy. Digital Guardian opened a window to tell us the action was denied; however, the console opened anyway.

The Digital Guardian policy denies access to files by specifying the path name in the form of a regular expression. We found that this shell could get around any Digital Guardian restrictions that look purely at the path name, e.g., we could rename files with the *.mdb* extension, or modify them directly.

The Digital Guardian policy keeps applications from executing by specifying either the filename, or the cryptographic hash (MD5) of the application's binary. We found that the rules specifying hashes still applied; however such rules that explicitly deny execution can be circumvented via other means (see Issues 14.7.6 and 14.7.7).

Prerequisites: An attacker requires access to a user account with administrative access and less than one minute to gain *SYSTEM* access.

Impact: Digital Guardian protections can be circumvented, and the election database can be compromised.

Procedural Mitigations: None.

Verification: This issue was confirmed via demonstration with the GEMS server.

14.7.6 The Digital Guardian policy denies only specific known unwanted applications

The Digital Guardian policy protects the system from modification by denying specific applications from executing. An attacker can make simply modifications those applications to allow them to run.

Description: The Digital Guardian policy protects the system from modification denying specific applications from execution. This technique is commonly know as blacklisting. The policy covers many “dangerous” applications such as registry editors and system management control panels that administrators typically use to reconfigure a system. Fundamentally, many recommend against the use of blacklists, because enumerating *every* potential bad item, in this case application, is often impractical. Issue 14.7.5 describes such an application that was missed by the policy designers. Furthermore, using a blacklist in this way cannot stop applications copied to the machine by an attacker.

The current Digital Guardian policy uses two techniques to specify blacklisted applications. The first technique specifies a filename. Whenever the filename appears in a file system operation, e.g., read, copy, and execute, the operation is denied. Unfortunately, an attacker wishing to overcome this restriction can bring along the same application with a slightly different name.

The second technique specifies the cryptographic hash (MD5) of the application binary. Applications matching the blacklisted MD5 hashes are denied execution; however, the Digital Guardian implementation does not operate exactly like this, see Issue 14.7.7. Again, an attacker can overcome this restriction. The rule does not keep the attacker from copying the executable and then slightly modifying internal text fields with a file (hex) editor. This will change the MD5 hash and the application will no longer match the blacklisted

value.

Circumventing the blacklist restrictions can manifest itself in many ways. The blacklist exists to maintain the system's configuration. By changing the system configuration an attacker changes the way the Digital Guardian policy is interpreted. For example, election procedure can dictate that the *GEMSAdmin* account (used to manipulate election databases) is disabled to restrict how the election database is modified. An attacker can bypass blacklisted applications to temporarily re-enable the account long enough to make desired changes.

Prerequisites: Depending on applications existing on the GEMS server, an attacker may require the ability to copy files to machine, e.g., CDROM, USB, network. With the proper tools, the Digital Guardian policy can be circumvented within seconds.

Impact: Digital Guardian policy can be circumvented, potentially enabling an attacker to disable Digital Guardian or modify the election database.

Procedural Mitigations: None.

Verification: This issue was confirmed via demonstration with the GEMS server and provided documentation of the Digital Guardian policy.

14.7.7 Digital Guardian execution restrictions are circumventable by copying files

The Digital Guardian policy protects the system from modification by denying specific applications from executing. Flaws in the way Digital Guardian identifies applications allows an attacker to simply copy a restricted application in order to run it.

Description: As discussed in Issue 14.7.6, the Digital Guardian policy blacklists a number of potentially dangerous applications by specifying the cryptographic hash (MD5) of the binary executable. While Issue 14.7.6 discussed how this technique is trivially circumventable using a file (hex) editor, there is a flaw with the way Digital Guardian matches an application to a blacklisted MD5 hash value. By specifying MD5 hash values in the Digital Guardian policy, one would expect the hash value to be calculated for every application that executes; however, this is not the case.

When a blacklisted application is copied to a local directory, Digital Guardian allows the program to execute. We found that after the system ran for some duration and was rebooted, possibly multiple times, Digital Guardian spontaneously began restricting the execution of the application copy. We cannot be sure of exactly how Digital Guardian functions without access to the source code.

While we did not fully investigate the situations under which Digital Guardian begins restricting the application copy, we performed a few experiments to gain a better understanding. We found that modifying the application copy with a hex editor (as described in Issue 14.7.6) keeps Digital Guardian from identifying the copy, and its execution is never restricted. We found that modifying an application copy that was already found by Digital Guardian with a hex editor allows it to execute.

Prerequisites: Access to a user account on the GEMS server.

Impact: Users can execute applications that are denied by the Digital Guardian policy.

Procedural Mitigations: None.

Verification: This issue was confirmed via demonstration with the GEMS server.

14.7.8 *GEMSUser* can use the CD burning application to modify the election database

One goal of the Digital Guardian policy is to only allow GEMS to modify election databases. A policy decision was made to allow *GEMSUser* to make CD backups of the election database. The *GEMSUser* can use the CD burning application to rename the election database, modify it using an arbitrary application, and restore the file to its original name.

Description: One goal of the Digital Guardian policy is to only allow *GEMSUser* to interact with election database files using GEMS. The only purpose of the *GEMSAdmin* account is to manage database files with `explorer.exe`. However, on election day, *GEMSUser* must frequently backup the election database. The current Digital Guardian policy allows *GEMSUser* to interact with `.mdb` and `.gbf` files using the Nero Burning ROM application to back up the files to a CD-R. Note, `.mdb` files contain the election database modified by GEMS, and `.gbf` files contain an encrypted copy of an election database; however Digital Guardian policy treats both file types identically.

Unfortunately, the file browser included with the Nero Burning ROM allows GEMS to copy and rename files. Using Nero, an attacker with access to the *GEMSUser* account can rename `.mdb` files to a different file extension to gain unmediated access. Additionally, note that the Digital Guardian policy has an explicit rule to keep GEMS from deleting `.mdb` files; however, Nero is not included in this rule.

Prerequisites: Access to the *GEMSUser* account.

Impact: The *GEMSUser* can gain unrestricted access to election databases.

Procedural Mitigations: None.

Verification: This issue was confirmed via demonstration with the GEMS server.

14.7.9 The election database protections only apply to files on the local hard drive

One goal of the Digital Guardian policy is to only allow GEMS to modify election databases. However, the current policy does not apply to files that are not on the local fixed hard drive. For example, if an election database file is copied to a USB flash drive, arbitrary applications may modify it. This allows an attacker to modify copied election databases with an application the depends on the file extension.

Description: One goal of the Digital Guardian policy is to only allow *GEMSUser* to interact with election database files using GEMS. The Digital Guardian policy specifies which applications may access the `.mdb` and `.gbf` election database files. However, the rule only applies when the files reside on the local fixed hard drive (explicitly stated in the policy definition). As a result, the policy allows unrestricted access to `.mdb` and `.gbf` files on external media, e.g., a USB flash drive or CDROMs. Combining this issue with Issue 14.7.8, an attacker can use Nero to copy a `.mdb` file to a USB flash drive, and then use Issue 13.1.2 to modify the file before copying the file back to the GEMS directory with Nero. While this issue may appear to provide only minimal advantage over Issue 14.7.8, we note that the attacker may wish to use a database tool that requires a file to be of a known file extension.

Prerequisites: An attacker requires a few minutes of access to the *GEMSUser* account, an attached USB flash drive, and Issue 14.7.8 or some other way of copying the `.mdb` files, e.g., Issues 14.7.2 and 14.7.5.

Impact: The *GEMSUser* can modify `.mdb` files copied to a USB flash drive.

Procedural Mitigations: None.

Verification: This issue was confirmed via demonstration with the GEMS server.

14.7.10 Digital Guardian logging is disabled

The instructions for installing Digital Guardian specify that logging should be disabled. As such, Digital Guardian does not report attempts to circumvent it, despite the displayed message indicating an infraction has been logged.

Description: Digital Guardian provides additional protection of the GEMS server. When a user performs an action that is denied by the Digital Guardian policy, that action is denied and a dialog box informs the user that the action has been blocked and recorded. However, installation guides provided for setting up Digital Guardian on GEMS servers explicitly indicates that the “Enable Activity Detail Logging” option should be unchecked. This guide corroborates discussions with state employees indicating Digital Guardian logging is disabled due to storage concerns. Furthermore, we were unable to view policy violations in the Digital Guardian console laptop used to administer the GEMS servers. Finally, we note that system security logs often contain many false positives. We cannot speak to the usefulness of the logs produced by Digital Guardian (having never seen them).

Prerequisites: None.

Impact: Digital Guardian does not report policy violations or other attempts to circumvent restrictions.

Procedural Mitigations: None.

Verification: This issue was confirmed by document analysis and via demonstration with the GEMS server.

14.7.11 Digital Guardian does not immediately detect if GEMS is replaced

Flaws in the way Digital Guardian identifies applications allow an attacker to replace the GEMS application and gain unrestricted access to election databases. We were unable to conclusively confirm the duration before Digital Guardian detects the change, if ever. It is possible that GEMS could be replaced with a malicious application for the duration of an election.

Description: One goal of the Digital Guardian policy is to only allow *GEMSUser* to interact with election database files using GEMS. As discussed in Issue 14.7.7, there is a flaw in the way Digital Guardian identifies applications. Issue 14.7.7 described how the Digital Guardian policy identifies applications that may not run by specifying the cryptographic hash (MD5) of the application’s binary executable. The vulnerability in Issue 14.7.7 is that an attacker can copy restricted applications in order to run them.

The Digital Guardian policy uses the same hash value identification in order to specify that *GEMS.exe* is one of the only applications that may access election database files (*.mdb* and *.gbf* files). In the system’s current configuration, an attacker can copy a malicious program to the exact file path of the GEMS executable, and even though the MD5 hash of the malicious application is different from that specified in the policy, Digital Guardian treats it as if it was GEMS.

We were unable to conclusively confirm if Digital Guardian will eventually recognize the new application has a different hash value. In our limited testing, Digital Guardian did not recognize the application replacement. However, regardless of whether or not the malicious application is recognized, the issue gives an attacker a sizeable window to mount an attack.

Prerequisites: An attacker requires access to a user account on the GEMS server that has permission to modify or rename the GEMS executable.

Impact: Any administrative user, including *GEMSUser*, can replace the GEMS application with a malicious application to gain unrestricted access to election databases. We were unable to conclusively confirm how long the GEMS application can be replaced before Digital Guardian detects the change. It is long enough to mount an attack that immediately modifies the database; however, there is a chance Digital Guardian will not detect the change long enough to allow the malicious application to remain for the duration of an election.

Procedural Mitigations: None.

Verification: This issue was confirmed via demonstration with the GEMS server.

14.8 Premier AV-TSX Vulnerabilities

14.8.1 The wires connecting the VVPAT printer can be cut without opening the AV-TSX

The power and data cables connecting the printer to the AV-TSX are easily accessible without opening the casing.

Description: In the state of Ohio, the paper receipt produced by the *Verifiable Voter Paper Audit Trail* (VVPAT) printer is the official record for an election. Accordingly, a machine may not be used in an election if its printer is not properly functioning.

The plastic enclosure surrounding the printer is highly flexible. By simply pushing the bottom edge of this enclosure, the data cables and power supply connecting the printer to the AV-TSX become exposed. The space created by pushing the plastic can easily fit a number of fingers or a small cutting device. An attacker could therefore successfully disable the printer in a matter of seconds.

Prerequisites: A voter or poll worker with 30 seconds of access to the machine could exploit this vulnerability.

Impact: If voters were not aware that a paper receipt was supposed to be printed, an unverifiable election may be run on the attacked unit. Should an audit expose that the printer was not working, the votes cast before this discovery may need to be thrown out. The attack could also be used to launch a Denial of Service, preventing this machine from being used in a polling place. Polling places with a small number of machines may be completely shut down by one or a few attackers.

Procedural Mitigations: Tamper evident seals are a potential partial mitigation to alert poll workers that an attack has occurred. For practical limitations, see Section 3.5. This particular attack, however, is obvious by its nature as the printer will no longer work.

Verification: This vulnerability was confirmed through penetration testing.

14.8.2 The plastic housing protecting the printer can easily be removed

The printer is attached to the AV-TSX by a protective plastic enclosure. However, this cover is easily removable, giving an attacker access to the records of previously cast ballots.

Description: The VVPAT printer is enclosed in a plastic housing. When set properly over the printer, the enclosure is secure to the voting machine using two mechanisms. The first, a 1/8 inch plastic latch, sits below the spool of cast votes. Access to this latch is protected by a lock which keeps the cover to the printer closed.

Without opening the cover and accessing the latch, the enclosure protecting the printer can be removed. An attacker would then have direct access to the printer, all of the paper spools and the wiring inside.

Prerequisites: A voter or poll worker with 30 seconds of access to the machine could exploit this vulnerability.

Impact: Like the previous exploit, this attack would allow an attacker to render the printer useless. More critically, the attacker could gain access to the unused paper (potentially modifying it to cause jamming errors) or even gain access to cast ballots. A voter or poll worker could successfully carry out this attack with under 30 seconds of access to an AV-TSX.

Procedural Mitigations: Tamper evident seals are a potential partial mitigation to alert poll workers that an attack has occurred. For practical limitations, see Section 3.5. This particular attack, however, is obvious by its nature.

Verification: This vulnerability was confirmed through penetration testing.

14.8.3 An adversary can destroy paper copies of cast ballots without opening the AV-TSX

The plastic enclosure protecting the printed paper ballots is not adequately sealed. An adversary can press against or inject a number of common household chemicals into the enclosure and destroy all paper records.

Description: The plastic enclosure protecting the VVPAT printer allows a voter to see the paper copy of their ballot without allowing them to directly access that paper. However, both the clear plastic window and the enclosure itself are not sufficiently sealed. An attacker can exploit this weakness by using a syringe to inject any number of compounds known to degrade/destroy information written to thermal printer paper.

Because of the design of the AV-TSX, it may be possible to damage the ballot paper trail in multiple ways. First, such a compound could be inserted in multiple ways such that all previous paper ballots stored in a machine would become unreadable. Alternatively, all of the unused paper in the machine could be attacked, preventing all future votes from being tallied. An example of this attack is shown in Figure 14.3.

A similar vulnerability was discussed in the California Red Team report (Issue 4.f); however, the details of this vulnerability were not listed in the public report.

Alternatively, the printer can be caused to jam simply by pressing against the enclosure. All results and logs sent to the printer after this occurs print over each other and are therefore unreadable.

Prerequisites: An attacker could successfully execute this attack with a few seconds of access to the AV-TSX.

Impact: An attacker would be able to damage or destroy the current and future votes cast on a roll of paper stored in an AV-TSX.

Procedural Mitigations: None.

Verification: This vulnerability was discovered through red teaming analysis.

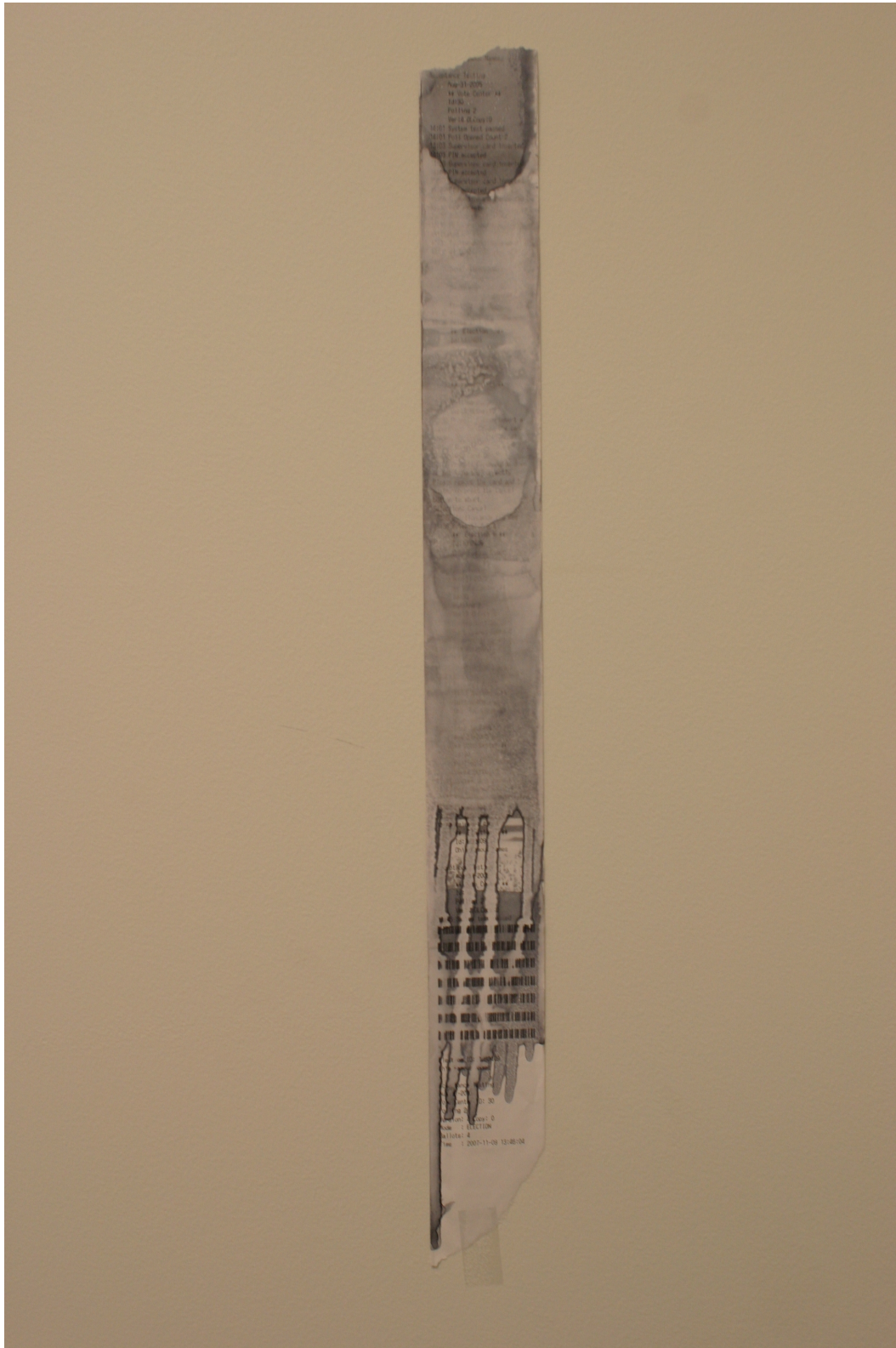


Figure 14.3: A paper copy of the system log destroyed by injecting a household chemical into the AV-TSX. No seals were broken in this attack.

14.8.4 The power button is accessible when all panels on the AV-TSX are closed and locked

The AV-TSX power button is located behind the same locked door protecting the memory card. Because this door is not sufficiently sealed, an attacker can easily turn the power on and off.

Description: The power button is located next to the PCMCIA memory card slot on the upper left hand side of the machine. Both of these parts of the machine are protected by a locked plastic shield when the AV-TSX is used in an election. However, the placement of the power button and the gapping between the shield and the main chassis allow an attacker to turn of an AV-TSX without removing the shield.

Prerequisites: A voter or poll worker with 30 seconds of access to the machine could exploit this vulnerability. Members of our team were regularly able to exploit this issue in under 10 seconds.

Impact: An adversary could cause a machine to shut down unexpectedly. This is a necessary prerequisite to a number of other attacks.

Procedural Mitigations: None.

Verification: This vulnerability was confirmed through penetration testing.

14.8.5 The bootloader can be manipulated to give access to the file system on the AV-TSX

The bootloader will allow an attacker to run Windows Explorer on the AV-TSX if a conditional statement evaluates to “True”. This value is set to false in commercial releases; however, simply making one change to the binary file will allow Windows Explorer to run.

Description: Previous versions of the bootloader software allowed an attacker to run Windows Explorer instead of BallotStation simply by including the file `explorer.glb` on a memory card. This method of running Windows Explorer was removed from the version of the code we examined.

However, it is still possible for the AV-TSX to boot Windows Explorer instead of BallotStation. By setting a variable system, an attacker with access to a binary of the bootloader can use software called a “hex editor” to gain access to Windows Explorer simply by changing the value of the variable. This attack is possible because the code to run Windows Explorer is included in commercial releases of the build.

An attacker could gain access to the bootloader in any number of ways. As an insider, they may be able to take the binary from elections headquarters. A poll worker with access to a machine during a “sleep-over” may be able to read the binary from the AV-TSX by exploiting Issue 13.3.4. States may also potentially sell older or surplus voting units, giving the general public access to the machine and a valid bootloader.

Having acquired a copy of the binary bootloader, the attacker could then change the value of the variable to TRUE using a hex editor. This change would be relatively quick to make. The attacker could load the modified bootloader onto memory cards and, because of Issue 13.3.1, cause any machine to load Windows Explorer.

Note that there is no need for Windows Explorer to be run on a commercial build of the software. Premier has built in a number of software update mechanisms into the memory card itself (all of which are insecure - see Issues 13.3.1 and 13.3.2). At no time should anyone be able to run Windows Explorer in the polling place.

Prerequisites: An attacker would need access to a copy of the bootloader binary and a hex editor.

Impact: An adversary could use Windows Explorer to change arbitrary settings and install malicious software on the AV-TSX. This would provide a helpful user interface such that the attacker creating the modified bootloader could rely on less sophisticated attackers to exploit the vulnerability in polling places.

Procedural Mitigations: No election official should be given unsupervised overnight access to equipment such as the AV-TSX (known as a “sleep-over”). Tamper evident seals are a potential *partial* mitigation to denote access to the memory card slot has occurred. For practical limitations, see Section 3.5.

Verification: This vulnerability was confirmed through source code analysis.

14.8.6 The memory of an AV-TSX can be erased by a memory card

While booting, the AV-TSX checks for the presence of a number of files. If these files are present, it will delete the contents of its registry and/or entire memory. The machine would not be usable until a legitimate update was reinstalled by election officials.

Description: Before BallotStation is started, the bootloader searches a memory card for a number of files. As discussed in Issue 13.3.1, this mechanism is designed to allow software updates to propagate to the AV-TSX. When checking to see if a new bootloader or operating system are present on the memory card, the bootloader looks for two additional files: `ERASEPSM.STL` and a second previously unnamed file (see the Private Appendix). If files with these names are present on the memory card, the AV-TSX will delete the contents of its entire flash memory or just the contents of the registry, respectively. This is achieved by a call to `FlashErase`, which overwrites the contents of the memory with `0xFF` (a string of ones).

Part of the vulnerability was identified by Hursti⁵. In previous versions of the software, Hursti noted that the presence of the file `ERASEPSM.STL` on a memory card caused the AV-TSX to delete the contents of its flash memory. While the current version of the bootloader contains a similar issue (with the filename slightly modified), it now requires the `DEBUG` jumper (Issue 13.3.4) to be set in order to delete the entire memory. As shown earlier, however, this would do little to stop a determined adversary. The registry, however, can be deleted without any requirements beyond the presence of the second previously unnamed file.

Neither file is authenticated in any way. Accordingly, anyone that can access a memory card for a few seconds can write the necessary files.

Part of this vulnerability (flashing the contents of the entire memory) was previously reported by Hursti⁶. Flashing the registry alone has not previously been reported.

Prerequisites: An attacker would need access to the memory card for both attacks. To delete the entire memory, the attacker would also need to be able to set the `DEBUG` jumper (see Issue 13.3.4).

Impact: An attacker would not need to delete the entire contents of the memory in order to prevent the AV-TSX from operating. Because so many critical values are stored in the registry (e.g., cryptographic keys, machine serial number), deleting the registry would prevent the AV-TSX from operating correctly. Because a new registry is included with the BallotStation binary, loading a new `.ins` file could reverse this attack if it were launched at the polling station.

Allowing poll workers to have access to a memory card and to the memory card slot in public may be more dangerous. This card may easily be misplaced or stolen by an attacker, giving them the ability to develop more sophisticated attacks against the system. However, in the absence of a card with a valid build of

⁵Harri Hursti, *Supplemental report, additional observations*. Black Box Voting, Unredacted release July 2, 2006, May 22, 2006.

⁶Hursti, ‘Supplemental report, additional observations’ (as in n. 5).

BallotStation, the exploited AV-TSX would be unusable.

Procedural Mitigations: Chain of custody logs are a potential *partial* mitigation to protect memory cards. For practical limitations, see Section 3.5.

Verification: This vulnerability was confirmed through source code analysis.

14.8.7 A voter can gain administrative access to the AV-TSX

Any voter can discretely gain access to the administrative screens of an AV-TSX. Among other options, this attacker can unload or delete the results of an election.

Description: Voters necessarily have limited capabilities on the AV-TSX voting machine. Many operations, such as unloading an election, printing reports and manipulating settings must only be accessible by trusted elections officials with access to the correct smart cards. However, it is possible for a voter to gain central administrator-level access to a machine during the course of an election. Most critically, votes can delete all cast ballots stored on both the memory card and internal memory without any special tools.

We discuss the details of this attack in the private appendix.

This attack was independently discovered; however, at the end of our analysis, we were given access to the private reports from the California TTB review. These reports indicate a similar attack yielding the same outcome.

Prerequisites: With any smart card and the ability to turn the power on and off, the attacker can achieve this elevated level of access.

Impact: An attacker could delete the results of an election or make an AV-TSX unusable.

Procedural Mitigations: None.

Verification: This vulnerability was confirmed through penetration testing.

14.8.8 A voter can cast an unlimited number of votes without any tools or knowledge

A voter with no special tools can program an infinite number of smart cards and therefore cast an unlimited number of votes. The attacker does not need to break any protocols or intercept any communications.

Description: A Voter smart card allows each voter to cast a single ballot when they interact with the AV-TSX. However, without any special tools or the Supervisor smart card, it is possible for a voter to create an unlimited number of voter cards using the AV-TSX. Alternatively, the attacker can reprogram a single smart card an unlimited number of times.

This attacks causes the machine to incorrectly enter the Supervisor Menu, from which the user is given the option to create voter cards.

Prerequisites: The attacker simply needs access to the AV-TSX during an election. Further details are contained within the Private Appendix.

Impact: An attacker can cast an unlimited number of votes without any special tools or previous interaction with any AV-TSX.

Procedural Mitigations: None.

Verification: This vulnerability was confirmed through penetration testing.

14.8.9 The smart card reader can be jammed by an attacker

An attacker can cause the card reader on the AV-TSX to hold a smart card, thereby blocking anyone else from using the machine. This attack is temporary and can be mitigated by rebooting the machine.

Description: When the smart card reader on the AV-TSX is finished reading a valid card or recognizes that an invalid input has been inserted, it attempts to physically eject the card from the slot. This allows other voters or administrators to apply valid cards and gain appropriate access to the AV-TSX.

It is possible, however, to cause the AV-TSX to refuse to return a smart card to a user. In so doing, subsequent voters and or administrators will not be able to use the machine as intended as they can not insert their valid cards. Because the majority of the card is kept inside the machine, even a reasonable amount of force will not dislodge the card. The card can be removed with a pair of pliers; however, this causes the entrance of the card reader to close, thereby preventing other cards from being inserted.

The only means of freeing the card is by rebooting the machine. This will force a precinct administrator to unlock the compartment containing the power button and memory card during an election. Any tamper evident seal attached to this compartment will also have to be voided.

Prerequisites: An attacker requires no special tools or previous access to the AV-TSX.

Impact: An attacker can create a number of problems with such an attack. By jamming a card in the slot, an AV-TSX may become unusable for the remainder of an election. If this tactic is taken against a large number of machines in a precinct, administering an election may become difficult in the allotted time. Alternative, the attacker may wish to have an elections official intentionally void the tamper evident seal on the memory card compartment to remove any suspicion of subsequent attacks.

Procedural Mitigations: None.

Verification: This vulnerability was confirmed through penetration testing.

PREMIER COMBINED IMPLICATIONS AND ATTACK SCENARIOS

15.1 Casting an Unlimited Number of Ballots

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

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

We assume that our attacker approaches the voting booth during an election under normal circumstances. The attacker brings with them a stack of smart cards containing the default Smart Card Key (published on the Internet). After approaching the AV-TSX, the attacker begins by covering his/her tracks. Because the AV-TSX notes in its audit logs when cards have been encoded, the attacker accesses the Central Administrator by exploiting Issue 14.8.7. Here, the attacker can delete the contents of both the memory card and the AV-TSX, thereby erasing most evidence of the attack. To hide the card creation operations, the attacker then simply changes the time and date of the AV-TSX to a period before the election. This portion of the attack can be accomplished in just over one minute. Moreover, deleting the contents of the memory card and changing the time/date are not logged. Should the attacker also worry about the log information encoded on the VVPAT, they can exploit either weakness discussed in Issue 14.8.4.

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

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

15.2 Pre-Stuffing an Election

The concept of “pre-stuffing” an election using the AV-OS PC memory card is not new; however, the vulnerabilities that enable this attack are still present. Since the idea was first suggested by Hursti¹, subsequent researchers have demonstrated the attack and even proposed slight variations.

The AV-OS PC is the primary means of reading paper ballots at the polling place. The most critical component of the AV-OS PC is the memory card, which contains the number of races, the location of the ovals for candidates, the current number of votes for each candidate, and even “live code” used to tell the AV-OS PC how to print the Election Zero report.²

AV-OS memory cards have no integrity protection mechanisms (see Issue 13.2.1). An attacker that physically possesses the memory card can therefore modify it in arbitrary ways. For instance, an attacker gaining access to the memory card after an election can submit forged results. Before an election, an attacker can update the ballot coordinates for ovals and completely change a voters intention or even neutralize candidates such that they receive no votes. Researchers working for the Connecticut Secretary of State provided an extensive scenario under which such an attack may occur³. While this attack does not exactly “pre-stuff” an election, it definitely has significant impact on the outcome.

The original “pre-stuffing” attack proposed by Hursti makes use of lack of memory card data protection, the existence of “live code” on the memory card, and a flaw in the way the AV-OS PC counts and stores votes (see Issue 13.2.7). The AV-OS PC does not store individual votes; rather, it stores the aggregate votes for each candidate. Like personal computers, the AV-OS PC store numbers in fixed size storage, which means if a number is incremented past a maximum value, it will reset to zero. In the AV-OS PC, the maximum value is 65,535, and it does not check to see that value is ever reached. While 65,535 votes may seem sufficiently large for any polling place, an attacker can use the “roll-over” side-effect to “pre-stuff” an election such that one candidate has a fixed advantage over another. An attacker with physical access to the memory card before an election can adjust the vote counter for *Candidate A* to 65,526 (-10) and the counter for *Candidate B* to 10. This gives *Candidate B* an advantage of 20 votes, since after *Candidate A* receives 10 votes, the counter will hold the value of zero. Furthermore, at the end of the election, the total number of votes on the memory card will match identically with the number of ballots in the ballot box.

When the election official prints the Election Zero report, however, *Candidate A* and *Candidate B* will not be zero. With access to the memory card, overcoming this limitation is trivial. Because the AV-OS PC is instructed how to print the Election Zero report by running the “live code” on the unprotected memory card, an attacker can simply modify the “live code” for the Election Zero report to always print zero. See the California Diebold source code TTBR report (Issue 5.1.11) for a step by step walk through of how the vote counter “roll-over” allows this attack to occur.

Some claim that this “pre-stuffing” attack can be detected by using proper election procedure. The memory card is always in one of several modes (e.g., pre-election, election mode, post-election) as indicated by a value stored on the card itself. It has been suggested that waiting until the card is in the polling place before changing it to election mode will provide defense against this attack. However, in a study for the California

¹Harri Hursti, *The Black Box Report: Critical Security Issues with Diebold Optical Scan Design*. Black Box Voting, July 4, 2005 (URL: <http://www.blackboxvoting.org/BBVreport.pdf>).

²The Election Zero report is an important piece of documentation that assures poll workers and election officials that no votes have taken place before the election begins.

³A. Kiayias et al., *Security Assessment of the Diebold Optical Scan Voting Terminal*. UConn Voting Technology Research (VoTeR) Center, October 30, 2006 (URL: http://voter.engr.uconn.edu/voter/OS-Report_files/uconn-report-os.pdf).

Secretary of State, Wagner et al.⁴ determined that the attack can occur no matter memory card state during shipment (Finding 10).

Finally, a recent study for the Florida Department of State⁵ reviewed newer versions of the Premier source code than we were given. The study indicates that the “live code” now includes protections to keep an attacker from modifying it in transit. Without changing the “live code” on the memory card, the “pre-stuffing” attack can be detected before the election begins. However, the Florida study reported a flaw in the method used to protect the “live code,” making the attack still possible.

The only known detection for this “pre-stuffing” attack is to recount all of the paper ballots.

15.3 Voting Machine Virus

A virus is a malicious computer program with the ability to automatically move from machine to machine. Upon infection, such programs can change the contents of files, record a user’s operations or make a computer unusable. Because they can spread silently even in the presence of procedural safeguards, viruses represent a significant and realistic threat to the integrity of an election. By exploiting one of many of the issues presented in this report, an attacker could inject a virus into any AV-TSX, AV-OS, EMP or GEMS server and gain control of an election in a county.

We are by no means the first to suggest such vulnerability to viruses. Feldman et al.⁶ were in fact able to build their own virus targeting an older version of the Premier’s touchscreen voting machine (AccuVote-TS). As the memory card from the infected machine was inserted into other machines, these machines automatically became infected. Such a virus could easily be replicated by an attacker with moderate programming abilities and access to a voting machine.

Our review of the source code and penetration testing of the range of Premier machines indicates that such attacks are highly plausible. This report has not only confirmed many of the previously known vulnerabilities enabling such attacks, but it has also uncovered a number of new serious new avenues through which viruses could infect voting machines. We briefly discuss a few of the points through which such an attack could occur.

15.3.1 Infecting the GEMS Server

The GEMS server should be the best protected element of an election system. These systems run antivirus software, rely on Digital Guardian to restrict the operations a user can execute and are typically kept physically secure within a county’s election headquarters. However, none of these protections are sufficient to stop a virus targeting a voting machine.

Antivirus software can only recognize viruses that have been previously identified. Because voting machine viruses are not commonly found on the Internet, such virus “definitions” are not part of any known antivirus

⁴David Wagner, David Jefferson and Matt Bishop, *Security Analysis of the Diebold AccuBasic Interpreter*. Voting Systems Technology Assessment Advisory Board (VSTAAB), February 14, 2006 (URL: <http://www.ss.ca.gov/elections/voting-systems/security-analysis-of-the-diebold-accubasic-interpreter.pdf>).

⁵Ryan Gardner et al., *Software Review and Security Analysis of the Diebold Voting Machine Software*. Security and Assurance in Information Technology (SAIT) Laboratory, Florida State University, For the Florida Department of State, July 27, 2007 (URL: <http://election.dos.state.fl.us/pdf/SAITreport.pdf>).

⁶Ariel Feldman, J. Alex Halderman and Edward Felten, ‘Security Analysis of the Diebold AccuVote-TS Voting Machine’. In USENIX/ACCURATE Electronic Voting Technology Workshop (EVT). 2007.

software. Accordingly, such software offers little protection against viruses targeting the GEMS server. The Digital Guardian software is also circumventable. An attacker able to exploit the vulnerabilities detailed in this section could easily circumvent the protections provided by this software. Moreover, while physical security certainly eliminates a significant number of attacks, it simply does not prevent a virus from infecting the GEMS server. An insider, for instance, can easily bypass all physical protections and secretly inject a virus onto the GEMS server.

15.3.2 Infecting the EMP Server

The EMP server is used to encode the memory cards sent to precincts and tally the results on a card when an election is completed. As the intermediary between the GEMS server and AV-TSX units deployed in precincts, the EMP server has significant access to critical pieces of election materials. For instance, while the communications between the GEMS and EMP servers can be protected by encryption, an infected EMP server could change ballot definitions before loading them onto memory cards. At the end of an election, an infected EMP server could change the results of ballots and then forge the authenticity of all results using or generating the necessary keys (Issues 13.3.5 and 14.1.7). Many of the vulnerabilities found in the EMP server, some of which execute immediately upon the insertion of an infected memory card, would allow a virus to gain access to perform such attacks.

15.3.3 Infecting the AV-TSX

An attacker could load a virus onto an AV-TSX during multiple phases of an election using vulnerabilities such as Issues 13.3.1 and 13.3.2. If memory cards are placed in voting machines before delivery to a precinct, an attacker could use a number of known techniques to remove tamper evident seals and unlock the door protecting the memory card before the voting machine was delivered. Alternatively, a poll worker could access the memory card before an election as the power button is located behind the same locked door. In both of the above cases or if cards are delivered to precincts independently of machines, the memory card could be replaced by one containing a virus. When an election is finished and the memory card is inserted into the EMP server, the virus could continue to spread throughout the system.

15.4 Denying Voters the Ability to Vote

In many situations, preventing specific or all voters from participating in an election may be valuable to an attacker. For instance, the expense of attempting to run multiple elections may put significant financial strain on a county. Alternatively, by repeatedly “canceling” elections, an attacker may disenfranchise some portion of likely-voters. Vulnerabilities in every component of these systems allow such attacks to be successfully executed. We highlight a few of these in various devices.

Weaknesses in the ExpressPoll may allow an attacker to load their own modified version of the voter database (Issue 14.6.3). AV-TSX units can be crashed or made unusable through vulnerabilities including but not limited to Issues 13.3.18, 14.8.6 and 14.8.7. The EMP server can be made to crash or unusable through vulnerabilities such as Issue 14.1.4 and 14.1.10. GEMS can also be made to crash through weaknesses such as Issue 14.3.1.