Integrity of Electronic Voting Systems: Fallacious Use of Cryptography

Seda Davtyan Aggelos Kiayias Laurent Michel
Alexander Russell Alexander A. Shvartsman
Computer Science and Engineering Department, University of Connecticut, Storrs, CT 06269, USA and
Voting Systems Security, LLC, 1 Technology Drive, Tolland, CT 06084, USA
{seda,aggelos,ldm,acr,aas}@engr.uconn.edu

ABSTRACT

In recent years, electronic voting systems have been deployed in all U.S. elections. Despite the fact that cryptographic integrity checks are used in most such systems, several reports have documented serious security vulnerabilities of electronic voting terminals. We present an overview of the typical security and election vulnerabilities found in most, if not all, electronic election systems, and present a case study that illustrates such vulnerabilities. Our hands-on security analysis of the AccuVote TSx voting terminal — used by more than 12 million voters in over 350 jurisdictions in the U.S. — demonstrates certain new integrity vulnerabilities that are present in the system. We present two attacks based on these vulnerabilities: one attack swaps the votes of two candidates and another erases the name of one candidate from the slate. These attacks do not require modification of the operating system of the voting terminal (as was the case in a number of previous attacks) and are able to circumvent the cryptographic integrity checks implemented in the terminal. The attacks can be launched in a matter of minutes and require only a computer with the capability to mount a PCMCIA card file system (a default capability in most current operating systems). The attacks presented here were discovered through direct experimentation with the voting terminal and without access to any internal documentation or the source code from the manufacturer.

1. INTRODUCTION

The landscape of technology used in the elections in the United States has changed dramatically in recent years. The push to modernize election systems was motivated by the inadequacy of the older manual and electro-mechanical voting equipment and encouraged by the 2002 U.S. Help America Vote Act (HAVA). The various electronic equipment in use today is provided by several vendors and, unfortunately, in almost all cases the systems are inadequately designed to provide the crucially needed integrity and security guarantees. Integrity and security of voting systems became a national concern with the release of several reports documenting election system vulnerabilities. Among the first such reports, in 2005 H. Hursti [8] released his findings on the Diebold Optical Scan system (the so-called "Hursti Hack"). This was an early design that used only a superficial password protection to secure the system. Newer designs normally incorporate some cryptographic tools; however, the application of the tools remains haphazard. For example, in 2006 Felten [5] famously demonstrated the vulnerabilities of the Diebold Touch-Screen system despite its use of encryption.

An electronic voting system is a complex distributed system comprised of several types of devices, including (i) election management systems, (ii) electronic voting terminals, such as optical scan terminals, direct entry electronic terminals, and/or enhanced-access terminals for people with disabilities, (iii) voter-assist terminals, such as ballot marking devices, (iv) removable memory devices, such as memory cards, universal serial bus drives, compact flash drives, etc., (v) means of communication, including removable media, telephone and data networks.

Electronic voting terminals are complex computing devices that include sophisticated hardware and software. The behavior of any given voting terminal depends on the software/firmware pre-installed on the terminal, software/firmware installed as an upgrade, and software and data installed for the purposes of an election via removable media. Any such installation, including the installation of election-specific software and data via removable media, can completely change the behavior of the terminal. In particular, incorrect, incomplete, or even arbitrary precinct election results can be reported by a terminal due to errors or malicious interference.

Removable memory devices serve to deliver election configuration to electronic voting terminals and to convey the results to central tabulation. Such devices have proved to be a major source of vulnerabilities in electronic voting systems. The cards connect the election management system and the voting terminals into a large distributed system. Inadequate security measures (electromechanical, software, cryptographic, and physical custody) can allow errors, introduced inadvertently or as the result of deliberate tampering, to propagate through the entire system. Such errors can create broad tampering risks and lead, in extreme cases, to massive failures. Every component of such distributed electronic system is susceptible to attacks, both external attacks and insider attacks.

Although vendors improved their use of cryptography, the mere application of cryptographic mechanisms such as (i) hash checking for software integrity, (ii) encryption for confidentiality of election related data, and (iii) digital signatures for integrity of election data, does not guarantee in itself that the desired properties are achieved. Use of good tools must go hand-in-hand with good use of tools. In particular, severe security deficiencies have been reported in electronic voting terminals despite the use of cryptography. In this way, superficial uses of cryptography can lead to a false sense of security. Worse, cryptography can prevent mean-

SAC'12 March 25-29, 2012, Riva del Garda, Italy.

ingful independent technological audits of voting equipment when encryption obfuscates the auditable data. A vendor may provide its own test and audit tools, but relying on the self-test and self-audit features is problematic as one should never trust self-auditing software (cf. relying on a corporate entity to perform self-audit).

Contributions. In this paper we describe archetypal vulnerabilities inherent in the current generation of electronic voting machines, especially focusing on the vulnerabilities that are due to superficial uses of cryptography. We then present the results of our original case study that illustrate such patterns.

Our case study is based on an analysis of a "direct recording electronic" (DRE) voting terminal. This terminal was made available to a State for an independent evaluation to be performed by us. The terminal used in this study is the AccuVote TSx terminal manufactured by Premier Election Solutions. This terminal is deployed in over 350 jurisdictions in the U.S. that encompass over 12 million voters (VerifiedVoting.org). In our investigation we verify that there appears to be cryptographic integrity checking in the AV-TSx memory card. Nevertheless, we discover that the scope of the integrity checking is not as wide as it should have been. In particular, we find that in certain files that control the layout of the slate, the integrity checking is performed at the file level but not at the slate placement level. This flaw in the scope of the integrity check enabled us to modify the slate layout without triggering any alert from the terminal. Moreover, we found that when contents of slate components were invalidated the terminal did not issue an alert but instead chose to simply (and silently) ignore the corrupted file.

Based on the above vulnerabilities we designed and tested two attacks against the AV-TSx terminal. In the first, the attacker wishes to swap votes received by two candidates. The attacker can be successful provided that the sizes of the two files that define the candidate representation in the digital slate are identical. We found that is not a rare occurrence and in fact our test election contained such pairs of candidates. The swapping was applied to the name definitions of the two candidates and included the integrity check. In the second attack, the attacker simply wishes to make one of the candidates disappear from the slate. This can be achieved though a modification of the file that defines the layout of the candidate's name.

All our findings are based on straightforward experimentation with the voting terminal; we had no access to internal or proprietary information about the terminal or access to source code.

Given the above, the use of AV-TSx in an actual election becomes problematic. Indeed, the alterations of a card can be done with a PC with a PCMCIA slot. If this terminal is used in an actual election it is extremely important to keep the memory card sealed in place. Moreover, it is very important to modify the operating system with a comprehensive check of the memory card (but this can only be done with a comprehensive system upgrade).

We note that our terminal appeared to lack the exact bootstrapping vulnerabilities reported in [7] (but lacking access to any internal information or source code it is difficult to determine if the bootstrapping process is truly better secured now).

2. ELECTRONIC VOTING: SYSTEMS AND VULNERABILITIES

We now describe the overall technological landscape of electronic voting systems, then focus on the use of cryptography in electronic voting terminals and the specific security and integrity vulnerabilities associated with e-voting terminals that are due to incomplete or inadequate uses of cryptography.

2.1 Electronic Election Systems are Intrinsically Complex

The hardware in an electronic voting terminal is, in essence, general purpose computing equipment. For example, commodity Intel processors are used Premier's Accu-Vote, ES&S DS200, and the Avante VoteTracker. Other commodity hardware, such as USB interfaces, PCMCIA ports, ethernet ports, serial ports, or parallel ports are typical components in such systems (one or more of these is found in the DS200, VoteTracker, Accu-Vote, and ImageCast machines).

General purpose hardware can itself offer no guarantees as to the correctness of the vote processing: (i) Hardware itself can be faulty; even the hardware systems built under the most stringent quality control can be faulty, e.g., the infamous "Intel Pentium bug" that caused intermittent computation errors [6]. (ii) Alterations to the resident software can completely change the behavior of the machine despite the correctness of the hardware itself.

Additionally, most vendors also use off-the-shelf operating systems, such as Microsoft Windows, Windows CE and Linux. General purpose operating systems are truly staggering in term of complexity. (For instance, typical Windows and Linux systems are estimated to consist of well over 50 million lines of code, furthermore, Linux is written by thousands of volunteers worldwide.)

Despite a minimalistic interface presented to voters, a voting terminal is an extremely capable device comparable to personal computers in terms of complexity and are susceptible to similar weaknesses (e.g., viruses, malware, and unintentional errors).

While it is tempting to view a voting terminal in isolation, it is critical to view the entire system formed by hundreds (or even thousands) of voting terminals distributed over a large geographical area and ultimately interacting with a single central system, e.g., an election management system (EMS), for the preparation of the election and the tabulation of the results. It is therefore a large, complex distributed system (even if it is only sporadically interconnected, e.g., by means of programmed removable media devices). Where central aggregation of tallies is employed, showing that malicious exploits are impossible, and that computation and logic errors are not present, requires considering how the data from multiple voting terminals interacts with EMS.

Two observations are critical in this respect: (i) The safety and correctness of a large distributed system is only as good as its weakest link. Additionally, a single failure — whether benign or malicious — can ripple through and affect the entire system. (ii) Procedural counter-measures can be used to mitigate the weaknesses of the system, however, in a large system relying on many distributed procedural elements, the probability of a procedure failure can be extremely high, even if each individual procedure fails with small probability.

2.2 Use of Cryptography: Using Good Tools vs. Good Use of Tools

The cryptographic mechanisms typically used in conjunction with electronic voting systems include (i) cryptographic hash functions, (ii) encryption, and (iii) digital signatures. While these mechanisms are valuable, merely using them is not sufficient to ensure integrity of an electronic election system.

Cryptographic digital fingerprints (computed by *hash functions*) are used to check the integrity of a software module. A digital fingerprint is a short sequence of binary digits derived from and included with the module. Since it is extremely difficult to construct another software module with an identical fingerprint, digital fingerprints make it possible to check with high probability that the correct module is installed.

However, the mere employment of a digital fingerprint check does not necessarily guarantee that incorrect software module will be detected even if the used hash algorithm is standardized and believed to be secure, such as the Secure Hash Standard (SHS) [3]. To illustrate this point consider that a system running compromised software may deliberately try to misrepresent the hash value of its software image. If successful, rogue software can run undetected. To ensure that such attacks are thwarted it is imperative that the hash function calculation is guaranteed to be performed in a trustworthy fashion either through direct interaction with the target system's trusted hardware, or by using a trusted platform module (TPM) that can be relied on to perform the needed computation correctly.

Encryption relies on an algorithm and keys to hide information and prevent its recovery when the keys are unavailable. The keys themselves are pieces of information (sequences of binary digits) that control the behavior of the encryption and decryption algorithms.

Encryption does not necessarily guarantee confidentiality even if the encryption algorithm used is a standardized and believed to be secure algorithm, such as the Advanced Encryption Standard (AES) [1]. To illustrate this point consider a setting where AES is used to encrypt individual records. AES, on its own, does not guarantee that encrypting two identical records results in two distinct ciphers. As a result, applying encryption to a series of records that belong to a small set of possible forms does not prevent analysis of the resulting encrypted data, such as the data found on a removable memory card. This type of attack was in fact illustrated in the context of electronic voting systems [9].

Digital signatures are a mechanism for authenticating data records (such as messages, documents, database records). Digital signatures are analogous to hand-written signatures used for authenticating authorship. Specific algorithms using keys are used to produce signed digital data and subsequently to ascertain the authenticity of the data where it is to be used.

The mere employment of digital signatures does not necessarily guarantee integrity even if the signature algorithm is a standardized algorithm that is believed to be secure, such as the Digital Signature Algorithm (DSA) [2]. To minimize the risks of tampering, it is crucial to ensure that the signed data is interpreted correctly and is used as intended. For example, consider a direct-recording electronic voting terminal, where the digital ballot is a list of pairs of digitally signed records. The first element of the pair represents the candidate name and associated counter. The second element of the pair tells the terminal how the candidate's information is displayed for the voter on the screen. Note that while the records are signed the pairings themselves are not signed and can be tampered with. In principle, an attacker can use the absence of signature on the pairing to swap the representations of two candidates and therefore swap their votes. Such an exploit does not tamper with any signed data, but rearranges the data to induce an incorrect behavior. Based on the above, it is clear that cryptographic primitives used without a comprehensive security model does not guarantee impossibility of tampering and its advertising without specific details can lead to a false sense of security.

Cryptographic techniques can mitigate the risks of attacks against removable media cards. The level of protection depends upon the strength of the cryptographic techniques, upon the safe keeping of the digital keys used to protect the cards, but also upon the safe-keeping of the voting terminal themselves. Indeed, the firmware of the voting terminal necessarily holds a copy of the digital keys used to protect the removable media. A successful attack against the terminal compromises those keys that an attacker can use to produce forged, compromised removable media cards. This situation is analogous to one where a person always hides a physical key under the doormat – knowing where the key is hidden defeats the purpose of having a lock. The trust in the whole system depends on the vendor diligence in its engineering practices to produce firmware that make extensive and complete use of cryptographic techniques, on the vendor's dedication at safe-keeping all the digital keys, and with election officials to secure the voting terminals between elections.

2.3 Specific Vulnerabilities Pertaining to Electronic Voting Terminals

The functions of the voting terminal are controlled by firmware, including ballot processing, vote tallying, and tally reporting. Therefore, correctness is of paramount importance in assuring integrity of an overall election.

Most voting terminals are designed to be "upgradable" with new firmware versions through simple procedures where the new firmware is installed via a removable media. Any installation of new firmware results in essentially a new voting terminal whose functions may be completely different from the functions that existed prior to installation. Such installation must be viewed as completely invaliding any prior certification. Note that the existing firmware is responsible for validating the new firmware before installing it. This implies that the only entity in a position to certify that authorized firmware is installed is the vendor itself. If the validation itself is partial, or too weak, unauthorized firmware can slip through, be installed and take over the control of the entire machine (including every subsequent upgrade). Therefore, the trust in the whole system entirely rests on the vendor.

Vendors can use cryptographic techniques and digital keys to sign the new firmware. The old firmware is then responsible for checking the digital signature of the new firmware before installing it. These methods can minimize the risk of installing unauthorized firmware.

One Achilles' heel in using cryptographic techniques to protect against unauthorized firmware upgrade is that their effectiveness depends on the safe-guarding of the digital keys. If the vendor keys are exposed at any point, adversaries can impersonate the vendor and produce malicious firmware that appears legitimate. Once again, the trust in the whole system rests entirely on the vendor.

The removable media cards are used both for holding the description of the election (digital model of the ballot) and for holding the counters. Once a card is programmed on EMS, it is shipped to election officials to be inserted into the voting terminal where it stays for the duration of the election before being shipped back for aggregating the results (where central tabulation is used). The integrity of the card during the entire process is critical to the integrity of the election.

If the card can be tampered with while in transit to the precinct election officials, the entire system can be compromised. The election description can be made inconsistent with the paper ballot *leading to an incorrect interpretation* of the votes and therefore incorrect tallying. Malware can be copied onto the card and can be automatically installed when the media is inserted into the voting terminal. The malware can interfere with the firmware prior to and/or during the election to perturb the tallying. Worse, once the "infected" card returns to the election management system for aggregation, it can deliver its payload to EMS and compromise *all the media cards subsequently inserted* affecting the process on a much larger scale [5]. If the card can be tampered with while in transit after the election back to the election management system, the tallies it holds can be modified and malware can be injected as

well leading to the same large scale impacts, in the extreme case causing incorrect election results to be reported.

Thus it is imperative that any electronic voting system considered for deployment is evaluated by domain experts as a complete distributed system, and not only as a collection of standalone components.

The use of cryptographic techniques can increase the integrity of the electoral processes supported by electronic systems and make tampering more difficult. However inadequate, incomplete or incorrect uses of cryptography, and less-than-diligent or poorly designed management of cryptographic keys creates vulnerabilities and leads to a false sense of security.

Lastly, it is important to reiterate that it is critical and imperative to establish and enforce a suitable and secure chain of custody to minimize the risks of attacks or interference that can range from a simple denial of service (e.g., benign voting terminal malfunction or card destruction) to an elaborate tampering scheme designed to compromise elections in multiple precincts.

3. CASE STUDY: AV-TSx VOTING TERMINAL

In 2006 Felten [5] demonstrated vulnerabilities of an early version of the Diebold Touch-Screen (TS) system. The vendor stated in response that the examined system was "received from an undisclosed source" and that it "is not used anywhere in the country," concluding that "the study is unrealistic and inaccurate" (https://freedom-to-tinker.com/blog/felten/refuting-diebolds-response). In fairness to the vendor, the version of the terminal examined by Felten was obsolete.

In our case study we focus on the successor of the TS system, called AccuVote TSx (AV-TSx). We were asked by Connecticut Secretary of the State to examine an official release of the voting terminal obtained directly from the vendor. The terminal is shown in Figure 1. Our limited evaluation readily revealed that the system has in fact serious—previously undocumented—security vulnerabilities. It was shown that the effects of tampering with voting machines can be devastating, e.g., votes can be reassigned to arbitrary candidates, leading to invalid election results. Notably, our work was conducted without any access to vendor technical documentation.

AV-TSx is a Direct Recording Electronic (DRE) voting terminal. This refers to voting terminals that use a graphical user interface to let a voter record his intent directly in digital format. The tallying is performed internally by the terminal that maintains counters for each candidate and race. DRE terminals have been criticized for lack of verifiability. As a result many DRE terminals today employ a VVPAT (Voter Verified Paper Audit Trail) system: the terminal is equipped with a printer that produces a record reflecting the choices of the voter; the voter is supposed to verify the VVPAT record. After the election it is possible to perform a manual count using the VVPAT records.

The AV-TSx voting terminal was criticized in [7] and [12] due the following discovered security flaws:

(*i*) It was possible to relatively easily circumvent the bootstrapping process and modify the operational environment of the system; the absence of cryptographic checks in the bootstrapping process was identified in [7].

(*ii*) The key management was, by default, using a fixed hard-coded key (leaked on the Internet); this was identified in [12] where the importance of choosing fresh signing keys was highlighted.

Fixing these problems require changes in the boot-loading process as well as adherence to an appropriate key management practice to be followed by election officials. In [12], it was reported that the AV-TSx uses a cryptographic integrity check to make sure that the contents of the card was legitimate. The nature of the previously undocumented vulnerabilities that we discuss here concern this last security feature.

Figure 1: The AccuVote TSx voting terminal.

3.1 Security Vulnerabilities

We now discuss several security vulnerabilities of AV-TSx. The attacks presented in Section 3.2 focus mainly on the vulnerabilities associated with the memory card, however other identified issues ought to be investigated too.

3.1.1 Basic Characteristics of the System

The system used in this study included the following components: *AV-TSx* voting terminal: firmware version 4.6.4, bootloader version BLR7-1.2.1, Windows CE operating system version WCER7-410.2.1. GEMS software version 1.18 install on a laptop. Ethernet is used to connect these two systems. The GEMS software is used to manage the ballot information, load the election data onto the AV-TSx, and tabulate the results.

The memory card is a standard PCMCIA flash card with a FAT file system. The card contains the following file hierarchy.

```
/  (root directory)
   Election Data/
      N.xtr
      N.edb
      M.adt
      K.brs
   Trashcan/
```

Here N, M, and K are 32 hexadecimal digits (i.e., a 128 bit hex number). The `.xtr` file contains the election data information, the `.edb` file stores database information, the `.adt` file is the audit log, and the `.brs` file is the ballot box.

The election data file bundles many Rich Text Format (RTF) files to display candidate names, wave files for auditive assistance, images for the slate and information about the precinct. All these files are packaged together in a single `.xtr` file along with 128 bit integrity checks for each. Votes are encrypted using 128 bit AES and placed in the `.brs` file.

The AV-TSx hardware internal flash memory stores ballot information and voting results. It is used, for example, to accumulate results from several voting machines by repeatedly inserting their memory card.

3.1.2 Identified Vulnerabilities

We now summarize several AV-TSx system vulnerabilities discovered during our analysis.

Election Data and Database File.

While each candidate name is accompanied by a 128 bit integrity check, the terminal fails to use them effectively. A failed integrity check should render the terminal unresponsive. However, when the AV-TSx finds a mistmatch in the 128-bit integrity check, it silently omits the candidate on the slate, effectively removing him as an option.

The candidate names printed on the VVPAT record are based on the same RTF file that is displayed to the voter. However, the name printed for the final results is based on data from the `.edb` file. Because of this, voters could be unaware of any discrepancies between their cast votes and the internally recorded votes. Such a problem can only be detected by performing a manual count of the ballots from the VVPAT and comparing with the printed final counts.

Additionally, there is no global check to ensure the entire election data is correct. For example, two RTF files for distinct candidates can be swapped along with their integrity checks. A suitable global integrity check should catch such manipulation.

Electronic Ballot Box.

There appears to be no global cryptographic signature of the card contents. Without this, it may be possible to stuff the ballot box by creating a custom ballot box file. This may depend on insider information to obtain the correct AES key and ballot format, but could be a threat nonetheless. Any changes to the memory card outside the voting terminal should result in an error.

Upgrade Files and Backdoors.

As documented in [5], previous versions of the machine (TS) were susceptible to attacks through back door files. If present on the memory card, the machine would give the user full access to the OS, for debugging purposes. For TS machines it was documented in [7] that the back door files, with the different filenames, still exists and their processing at boot time occurs if the files have specific names. We remark that the bootstrapping process in our AV-TSx machine may still function as it is impossible to conclude positively that they are not working without having access to properly structured upgrade files. Yet, the filenames that worked for previous versions no longer seem to function and we were unable to discover any similar backdoors as yet (or to establish their absence). A similar threat exists for the upgrading mechanism. In previous versions, only the name of the upgrade file was used to identify a valid software upgrade located on the memory card. This represents a grave security vulnerability if no proper integrity checks are being used to authenticate the software upgrade. We did not have examples of legitimate upgrade files and could not assess this specific vulnerability.

Internal Storage Vulnerabilities.

The accumulation functionality requires inserting each memory card into a AV-TSx terminal so that the results can be merged with those already stored on the internal memory. Without source code, it is not clear how the AV-TSx determines the data to be merged. In particular, it is unclear whether or not a AV-TSx terminal could ship with a set of election results already present which could be merged with valid results.

3.2 The Attacks

The presented attacks were developed with precisely the same information and access to the system that is normally available to, for example, election administrators (supervisors, poll workers and other officials). To carry out the attack, one only needs physical access to the voting machine, without the privileges of an election administrator. It is important to reiterate that the attack development is based on straightforward experimentation with the voting terminal; we had no access to internal or proprietary information about the terminal or access to source code. An attacker only needs a few minutes with the card and a hex editor to perform the attack. In addition, an attacker may need to open the lock which covers the removable card. The attacker needs no knowledge of the particulars of the election he is to undermine (such as exact candidates' names, ballot layout, precinct names, or any kind of passwords). What the attacker needs is to find two `rtf` strings with the same length (first 4 bytes of the `rtf` string contain the `.rtf` file size) within the `.xtr` file. The whole process can be completed in a matter of a few minutes. In the following we give a step-by-step description of the attack.

3.2.1 Preliminaries

Any time a card is left unattended, or in transit without adequate chain-of-custody controls, it is vulnerable to tampering. Even if the AV-TSx terminal is locked within the ballot box prior to an election the memory card can be retrieved. If the box is unlocked or the attacker has the keys this is straightforward. The fact that the vendor appears to be using the same keys across multiple machines makes it easier to unlock the ballot-box (we had two terminals and they both shared the same keys). Note that the keys for these machines are difficult to copy because they are not standard size. Yet, a copy of this key is sent to every precinct and keys assigned to each location are not individually numbered, nor is there any record of which key is assigned to each precinct.

Once the PCMCIA card is accessible the attacker can have an immediate access to its contents through a commodity PCMCIA card reader.

3.2.2 The Details

Recall that the removable memory card contains four types of files: `brs, adt, edb`, and `xtr`. The attacks are concerned with the `xtr` file in which bundles an `rtf` file with the candidate name and instruction, an audio file (`wav`) and two bitmap representations (for end-user directions) for the slate. Each file is stored as follows:

4 bytes - filesize N; N bytes - data; 16 bytes - checksum.

The attacker needs to find two candidates for which the `.rtf` file sizes are identical and swap the corresponding `.rtf` files and checksums. If the checksums are not swapped, the data will not correspond to the checksum and the voting software will simply not display this entry (which is by itself a serious vulnerability).

A variant of the swapping attack simply *nullifies* the candidate name which triggers a silent suppression of the candidate from the slate.

The *Nullifying* Attack.

When the checksum is not consistent with the content of the `.rtf` file the AV-TSx terminal silently discards the candidate name. Thus, it suffices to flip a single bit in the data part of the .rtf file, without altering the length of the file, to achieve the desired effect. For example, we altered a candidate's .rtf file by replacing a 'C' with a 'D' resulting in a corresponding blank cell on the slate.

An example of the original untampered slate is given in Figure 2. The same slate after a candidate has been nullified is given in Figure 3. In all screen shots the last name of the candidates are blacked out and their first name is repeated white-on-black.

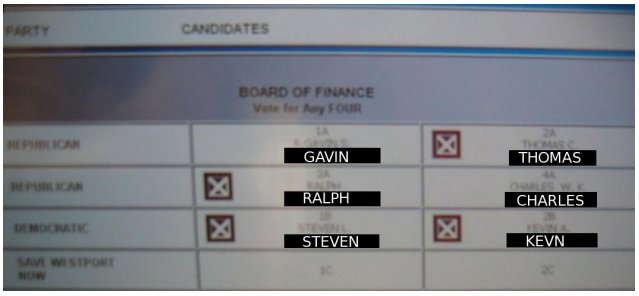

Figure 2: The original, untampered, slate. Some choices have been made by the voter.

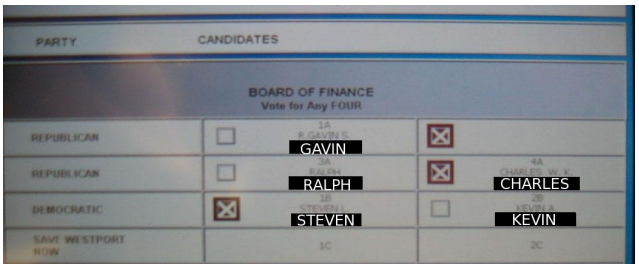

Figure 3: The slate with the nullified candidate name.

Voting proceeds as usual. When printing the ballot, if there were no votes for the (now blank) candidate, an entry is printed with no name for that candidate. For example, if we originally had the left print out, we now have the right one

...	...
[X] THOMAS C. THOMAS	[X]
...	...

When the election is finalized, the results are printed using the candidate's original name which reveals that the name is in fact stored in two places: (1) a label in a database record, and (2) within the formated `.rtf` file. Both appear in the GEMS database. Only the `.rtf` file is visible in the clear within the card contents. The database label must be either encrypted or compressed with other data. The database label is used on the zero report and the final report, while the `.rtf` file is displayed on the screen and printed on the paper ballot record. Interestingly this redundancy and normal looking record *help conceal the attack*.

Swapping Candidates.

This is accomplished by swapping the `.rtf` files of the candidates and corresponding checksums. We again held a two machine election, swapping the entries for one machine only. The slate presented by the untampered machine is given in Figure 4. Note that the lengths of the two `.rtf` files are identical since the name includes both the first and last names.

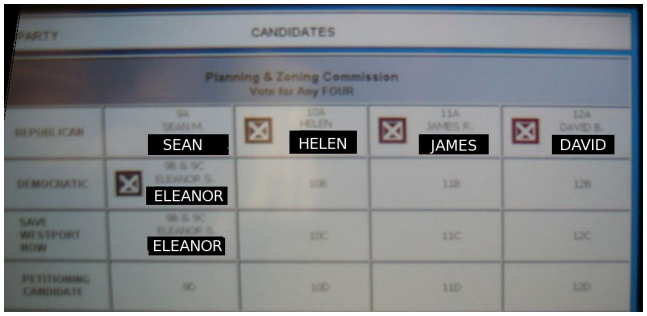

Figure 4: The ballot with unaltered candidates' names (before swapping)

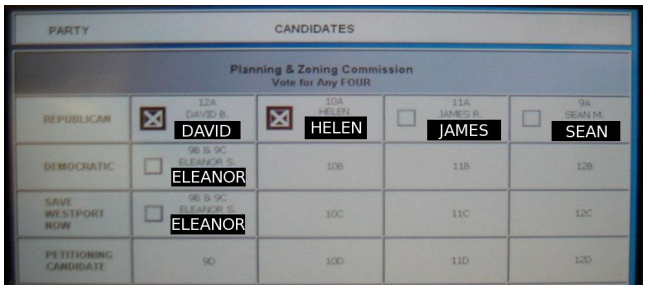

Figure 5: The ballot with swapped candidates' names

The tampered machine ran without an error, with the two candidates swapped. Figure 5 shows a screenshot with candidates "DAVID B. DAVID " and "SEAN M. SEAN " swapped.

We then voted twice for "DAVID B. DAVID ", on each machine (with the original and tampered elections loaded). The votes on the screen agreed with that on the printed VVPAT records (two for "DAVID B. DAVID ") in both cases (see the scans of the records in Figure 6, Figure 7).

The election ran correctly and a voter can verify that the printed record indeed corresponds to the choices made on the screen. However, the final results on the tampered machine showed two votes for "SEAN M SEAN " and zero for "DAVID B. DAVID " (Figure 8). On the untampered machine the printed ballots and the results match (Figure 7 and Figure 9).

We ran an election on two machines, with one of the memory cards tampered as described. Upon finishing the election, the results can be combined on AV-TSx with no reported errors. Namely, there is no consistency check to verify that the .xtr files match. Any votes for the blank spot are assigned to the candidate that originally should have appeared there.

We finally combined the results and send the tally to GEMS, with no errors. Figure 10 illustrates the aggregated results (of the tampered and untampered terminals) with two votes for each candidate "DAVID B. DAVID " and "SEAN M SEAN ", even though during the election no votes were given to "SEAN M SEAN ".

If an attacker has access to the memory card and two candidates have names of the same length, the attacker can swap their votes on that machine. Note, that the length requirement applies to the `rtf` files (not just the names) that also contain formatting such as spaces, newlines, and font information.

Completing the attack.

Once all the changes have been made to the `.xtr` file the mem-

Figure 6: Votes on the printed ballot (altered case)

Figure 7: Votes on the printed ballot (unaltered case)

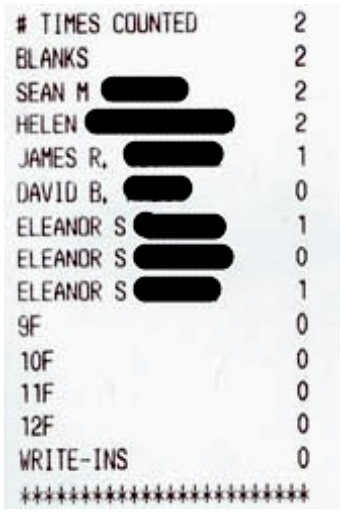

Figure 8: Results on a tampered machine

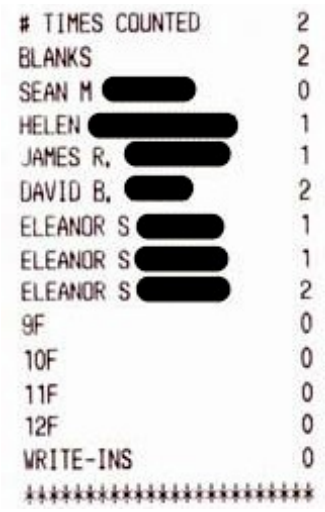

Figure 9: Results on unaltered machine

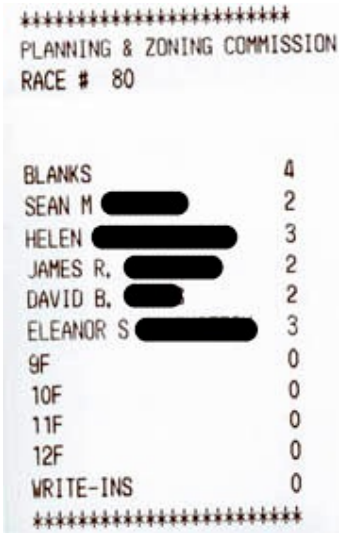

Figure 10: Aggregated results from both machines

ory card is ready for use. After this step, the AV-TSx terminal will be found by poll-workers in its expected pre-election state. The terminal will appear to be functioning normally for all operations during the election. The total time required to compromise the card is only a few minutes, depending on the dexterity of the attacker in picking the lock of the ballot box.

4. CONCLUSIONS

We presented typical security and integrity vulnerabilities found in the electronic voting equipment. To illustrate some deficiencies, we presented a case study using the AV-TSx system used in a large number of jurisdictions in the U.S. Specifically, we demonstrated two serious attacks against the integrity of the election process by exploiting inadequate and superficial use of cryptography in the target system. We stress that we did not possess the source code for the voting terminal or EMS. Compromising a terminal takes a few minutes using a commodity PCMCIA card reader and a hex editor. The conclusion is that great caution is warranted before employing AV-TSx in elections.

There have been several studies (e.g., [4, 10, 11]) that specifically addressed the issue of designing e-voting systems and offering recommendations for improvement. Here, we point out the particular shortcomings of the AV-TSx terminal and identify aspects that need to be dealt with to obtain a secure and robust system.

Global Integrity Check.

The memory card of the AV-TSx, a standard PCMCIA card holds the election data, ballot box, and the audit information. The major shortcoming that enabled our attacks is a lack of a global integrity check computed on the relevant contents of the card.

Modified Election Data Files and Integrity Checks.
The `.xtr` file contains the names of the candidates in RTF format. Each `.xtr` file does have a 16 byte integrity check. A failed integrity check should put the machine in an "insecure" state and cause an alert to be issued. However, the AV-TSx terminal fails to do this and simply omits that file when building the on-screen slate. A cryptographic check is ineffective if a failure is not suitably handled by the system.

Inconsistent File Usage.
The redundancy and lack of consistency check between the candidates name appearing in the `xtr` and `edb` files contributes to hiding the attack with a normal-looking printout during the initial testing by poll workers. The slate options displayed to voters should correspond exactly to the choices displayed on the final results.

Backdoor Files.
Previous versions of the machine were susceptible to attacks through back door files [5]. It is unclear whether similar backdoor still exist in the current AV-TSx; further investigation would be necessary in this regard.

Limited Software Accountability and Auditability.
There is no (documented) way to examine the software (Operating System) currently installed on the machine.

Our findings aptly demonstrate that merely using cryptographic tools may lead to a false sense of security. In order to be effective, cryptography must be used in conjunction with a sound design that provides comprehensive protection in safeguarding the integrity of critical information.

5. REFERENCES

[1] Federal Information Processing Standards Publication 197. Aes fips-197. National Institute of Standards and Technology, 2006. http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf.

[2] Federal Information Processing Standards Publication 197. Dsa fips 186-3. National Institute of Standards and Technology, 2009. http://csrc.nist.gov/publications/fips/fips186-3/fips_186-3.pdf.

[3] E. B. Barker and Q. Dang. Fips pub 180-3. National Institute of Standards and Technology, 2008. http://www.nist.gov/customcf/get_pdf.cfm?pub_id=901372.

[4] D. Chaum, P.Y.A. Ryan, and S.A. Schneider. A practical voter-verifiable election scheme. In *ESORICS*, pages 118–139, 2005.

[5] A.J. Feldman, A.J. Halderman, and E.W. Felten. Security analysis of the Diebold AccuVote-TS voting machine. In *USENIX/ACCURATE Electronic Voting Technology Workshop (EVT'07)*, 2007. http://www.usenix.org/events/evt07/tech/full_papers/feldman/feldman.pdf, also http://itpolicy.princeton.edu/voting/.

[6] T.R. Halfhill. An error in a lookup table created the infamous bug in Intel's latest processor. BYTE, 2005.

[7] H. Hursti. Diebold TSx evaluation, black box voting project, 2006. http://www.bbvdocs.org/reports/BBVreportIIunredacted.pdf.

[8] Harri Hursti. Critical security issues with Diebold optical scan design, July 4th, 2005.

[9] T. Kohno, A. Stubblefield, A.D. Rubin, and Wallach D.S. Analysis of an electronic voting system. In *IEEE Symposium on Security and Privacy*, pages 27–42, 2004.

[10] R. Mercuri. A better ballot box? *IEEE Spectrum*, 39(10), 2002.

[11] D. Molnar, T. Kohno, N. Sastry, and D. Wagner. Tamper-evident, history-independent, subliminal-free data structures on PROM storage -or- How to store ballots on a voting machine (extended abstract). *IEEE Security and Privacy*, 2006.

[12] D. Wagner, D. Jefferson, and M. Bishop. Security analysis of the Diebold AccuBasic interpreter. Voting Systems Technology Assessment Advisory Board, University of California, Berkeley, 2006.

Seda Davtyan is a PhD candidate in the Department of Computer Science and Engineering at the University of Connecticut. She received her M.S. degree in Informatics and Applied Mathematics (2003) and B.S. degree in Applied Mathematics (2001) from Yerevan State University. Her research interests include analysis, design and implementation of distributed and parallel algorithms, and evaluation of voting technologies.

Aggelos Kiayias is an Associate Professor of Computer Science and Engineering at the University of Connecticut. He is the head of the Crypto-DRM laboratory that is dedicated to the study of the cryptographic aspects of copyright technologies and digital rights management (DRM) systems. He is also a Principal Analyst at Voting Systems Security, LLC. Dr. Kiayias has been the recipient of an NSF Career award and a Fulbright fellowship. His research has been funded by a number of agencies including, NSF, DoD, DHS and NIST. He holds a Ph.D. from City U. of New York and is a graduate of the University of Athens, Greece.

Laurent Michel is an Associate Professor of Computer Science and Engineering at the University of Connecticut. He is also a Principal Analyst at Voting Systems Security, LLC. He received his M.S. and Ph.D. degrees in Computer Science from Brown University in 1996 and 1999 respectively. His interests spans Combinatorial Optimization with a particular emphasis on Constraint Programming, forecasting and voting technology. He has co-authored two monographs, more than 80 papers and sits on the Editorial Board of Constraints, Mathematical Programming Computation and Constraint Letters.

Alexander Russell Alexander is a Professor of Computer Science and Engineering at the University of Connecticut and a Principal Analyst at Voting Systems Security, LLC. He holds a Ph.D. in Applied Mathematics from Massachusetts Institute of Technology (1996), and a B.A. in Computer Science and Mathematics from Cornell University (1991).

Alexander A. Shvartsman is a Professor of Computer Science and Engineering and the Director of the Center for Voting Technology Research at the University of Connecticut. He is also a Principal Analyst at Voting Systems Security, LLC. Shvartsman holds Ph.D. from Brown University (1992), M.S. from Cornell University (1981), and a B.S. from Stevens Institute of Technology (1979), all in Computer Science. Prior to embarking on his academic career he worked as a member of technical staff at Bell Labs and Digital Equipment Corporation. His professional interests are in distributed computing, fault-tolerance, and integrity of electronic voting systems. He is an author of over 130 technical articles and three books.