How to steal an election by hacking the vote

by Jon Stokes

Version 1.01 – October 27, 2006

How to steal an election by hacking the vote

By Jon Stokes

What if I told you that it would take only one person—one highly motivated, but only moderately skilled bad apple, with either authorized or unauthorized access to the right company's internal computer network—to steal a statewide election? You might think I was crazy, or alarmist, or just talking about something that's only a remote, highly theoretical possibility. You also probably would think I was being really over-the-top if I told you that, without sweeping and very costly changes to the American electoral process, this scenario is almost certain to play out at some point in the future in some county or state in America, and that after it happens not only will we not have a clue as to what has taken place, but if we do get suspicious there will be no way to prove anything. You certainly wouldn't *want* to believe me, and I don't blame you.

So what if I told you that one highly motivated and moderately skilled bad apple could cause hundreds of millions of dollars in damage to America's private sector by unleashing a Windows virus from the safety of his parents' basement, and that many of the victims in the attack would never know that they'd been compromised? Before the rise of the Internet, this scenario also might've been considered alarmist folly by most, but now we know that it's all too real.

Thanks to the recent and rapid adoption of direct-recording electronic (DRE) voting machines in states and counties across America, the two scenarios that I just outlined have now become siblings (perhaps even fraternal twins) in the same large, unhappy family of information security (*infosec*) challenges. Our national election infrastructure is now largely an information technology infrastructure, so the problem of keeping our elections free of vote fraud is now an information security problem. If you've been keeping track of the news in the past few years, with its weekly litany of high-profile breaches in public- and private-sector networks, then you know how well we're (not) doing on the infosec front.

Over the course of almost eight years of reporting for Ars Technica, I've followed the merging of the areas of election security and information security, a merging that was accelerated much too rapidly in the wake of the 2000 presidential election. In all this time, I've yet to find a good way to convey to the non-technical public how well and truly screwed up we presently are, six years after the Florida recount. So now it's time to hit the panic button: In this article, **I'm going to show you how to steal an election.**

Now, I won't be giving you the kind of "push this, pull here" instructions for cracking specific machines that you can find scattered all over the Internet, in alarmingly lengthy PDF reports that detail vulnerability after vulnerability and exploit after exploit. (See the bibliography at the end of this article for that kind of information.) And I certainly won't be linking to any of the leaked Diebold source code, which is available in various corners of the online world. What I'll show you instead is a road map to the brave new world of electronic election manipulation, with just enough nuts-and-bolts detail to help you understand why things work the way they do.

Along the way, I'll also show you just how many different hands touch these electronic voting machines before and after a vote is cast, and I'll lay out just how vulnerable a DRE-based elections system is to what e-voting researchers have dubbed "wholesale fraud," i.e., the ability of an individual or a very small group to steal an entire election by making subtle changes in the right places.

So let's get right down to business and meet the tools that we're going to use to flip a race in favor of our preferred candidate.

***Note**: I'm not in any way encouraging anyone to actually go out and steal an election. This article is intended solely as a guide to the kinds of information and techniques that election thieves already have available, and not as an incitement to or an aid for committing crimes.*

E-voting 101: touch-screen machines and optical scanners

There are many different types of electronic voting machines available from a whole host of large and small vendors, but this article will focus primarily on one type: the direct-recording electronic (DRE) voting machine. Nonetheless, optical scanners are vulnerable to many of the same exploits that I'll describe for the DRE; the only difference is that optical scanners leave a reliable paper audit trail that could be used to tell if an election has been tampered with, but such audits must actually be carried out to have any impact.

DREs and optical scanners are far and away the two most popular types of voting machines in use today. The following statistics break down by popularity the types of voting machines used in 2006:

Voting equipment reported for the 2006 elections

Type of voting equipment	Counties		Registered voters*	
	Number	Percentage	Number	Percentage
Punch card	124	3.98	5,166,247	3.03
Lever	119	3.82	17,356,729	10.18
Paper ballots	176	5.65	653,704	0.38
Optical scan	1,502	48.23	69,517,991	40.79
Electronic	1,050	33.72	66,573,736	39.06
Mixed	143	4.59	11,154,765	6.55
Total	**3,114**	**100**	**170,423,172**	**100**

* Registered voter counts are from the November 2004 general elections
Source: Election Data Systems

Just to orient ourselves to the basics of electronic voting, let's take a brief look at how votes are cast and counted using each type of machine.

Optical scan machines

In order to cast a vote using an optical scan machine, a voter follows the three steps shown in Figure 1.

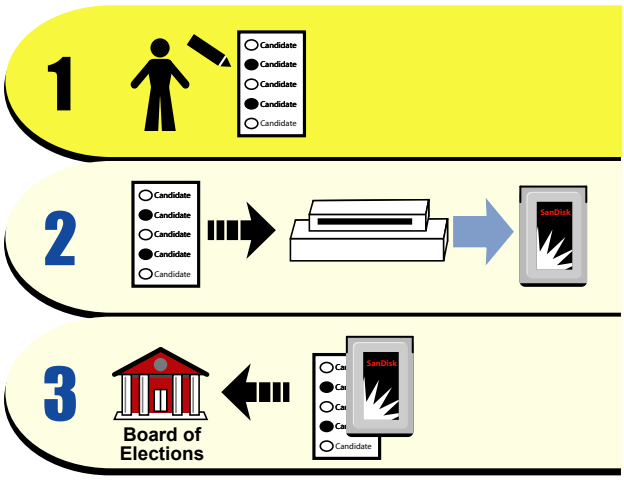


Figure 1: Electronic voting using an optical scanner

The three steps depicted in Figure 1 are as follows:

1. After receiving a paper ballot from poll workers, the voter marks her choices on the ballot by filling in bubbles with a pen. (An optical scan ballot looks and functions much like the multiple choice bubble sheets used in standardized tests.)
2. The marked ballot is then fed into the optical scan voting machine, where the voter's choices are translated in the 1s and 0s of computer language and stored, along with the rest of the votes cast on that machine, in the machine's internal memory. (I've depicted the internal storage as a SanDisk Flash PCMCIA card of the type commonly used in the Diebold DRE described below, but other storage formats are possible.)
3. At the end of the election, when all of the votes have been cast and are stored in the optical scan machines, the contents of the machines' internal storage devices are then transmitted to the county Board of Elections (BOE) for tallying and archiving. The marked paper ballots are also archived, in case a manual audit is demanded.

There are some variations in the process listed above (e.g., all of the votes in a single precinct can be tallied before being sent off to the BOE), but in general it describes overall movement of votes in the voting process.

Direct-recording electronic (DRE) machines

The steps involved in voting with DREs are similar to those described for optical scan machines, but there are some critical differences. Figure 2 illustrates what we might call the "life-cycle of a vote" in the DRE-based voting process.

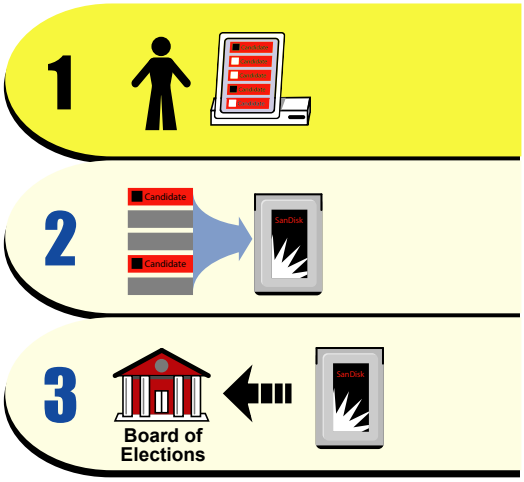


Figure 2: Electronic voting using a DRE

The steps depicted in Figure 2 are as follows:

1. The voter loads his ballot onto the DRE's screen by inserting into the machine the special smart card that he was issued by a poll worker. When the ballot screen appears, the voter marks his selections by touching the appropriate boxes on the screen.
2. The votes are read from the screen by the machine's vote recording software and recorded directly onto the DRE's internal storage, where they're stored along with the other votes that were cast on that machine.
3. At the end of the election, when all of the votes have been cast and are stored in the DREs, the contents of the machines' internal storage devices are then transmitted to the county Board of Elections (BOE) for tallying and archiving.

Note that the voter's choices are only recorded in one place: the internal storage of the DRE. Unlike the optical scan machines, the DRE system provides no permanent, nonelectronic paper record of the voter's intended choices that can be verified by the voter and then archived for possible use in an audit.

Now, the three-step process described above is vulnerable at multiple points in each stage. Here are just a few examples to illustrate what I'm talking about:

> **Step 1**: The machine could be tainted with vote-stealing software, or the voter could taint the machine with vote-stealing software by gaining access to it.
> **Step 2**: If the machine is tainted, then it can incorrectly record the vote. Or, if the voter has managed to make a supervisor card for himself, he can vote multiple times, delete votes, or disable the machine entirely.
> **Step 3**: If the centralized machine that does the vote tallying is tainted, then not only can it skew the election results, but it can also infect any DRE that connects directly to it, or it can taint any storage card that's plugged into it.

You might think that the supervisor smartcard cloning, viruses, and unauthorized accesses that I've described above are purely hypothetical. If the DRE in question is the popular Diebold AccuVote TS, then they're not at all hypothetical. All of the

attacks that I just summarized, and many more, have been implemented by multiple teams of security researchers. Just for kicks, take a break from reading and go watch this little demonstration video.

But before we talk in more detail about the AccuVote, let's take a step back and get a big-picture look at the kinds of new opportunities that the would-be election thief has at her disposal, thanks to DREs.

Bad apples and barrel sizes, or how to do a lot with a little

If we want to steal an election, then ideally we want as few warm bodies in on the scam as possible. All of the old-school election manipulation tricks, like voter intimidation, vote-buying, turn-out suppression, and so on, require legions of volunteers who know exactly what's going on; but in the new era of electronic vote tampering, an election thief can do a whole lot more with a whole lot less.

Election security experts break down voting fraud types into two main categories, based on how many bad apples it takes to swing an election: *retail fraud* and *wholesale fraud*. Retail fraud is the kind of election fraud that's most familiar to us, because it has been around for the longest time. In general, retail fraud involves multiple bad apples at the precinct level, carrying out any number of bad acts involving multiple voters and voting machines. Some examples of retail fraud are ballot stuffing, restricting polling place access by means of intimidation, vandalizing individual machines to make them unusable, counterfeiting ballots, and so on.

Wholesale fraud is relatively new, and it involves a single bad apple who can affect an election's outcome at the precinct, county, and state levels. (Actually, by this definition, wholesale fraud is as old as the poll tax. But let's stick to wholesale fraud involving electronic voting machines for now.) So with wholesale fraud, one bad apple can affect different barrels of various sizes, depending where in the election process she's placed.

The table below breaks down the newer types of fraud that electronic voting machines have made available to election thieves:

Wholesale and retail fraud

	Wholesale	**Retail**
Detectable	• Altering the vote tabulation process • Altering the record of tabulated results	• Multiple voting • Deleting votes • Disabling a machine • Invalidating all the votes on a machine
Undetectable	• Altering the vote tabulation process • Altering the vote recording process • Altering the record of votes	• Altering the vote recording process • Altering the record of votes

In this table, "detectable" denotes instances of tampering and fraud where we could potentially know that something went wrong with the vote, even if we're not sure

what has happened or how. Undetectable fraud denotes fraud that's absolutely impossible to detect after the fact (short of a whistleblower coming forth), and that's functionally impossible to detect before the fact due to time and resource constraints on pre-election machine testing.

The scariest part of Table 2's list of e-voting fraud types is the box where the "Undetectable" row and the "Wholesale" column intersect. Undetectable wholesale fraud is the ultimate apocalyptic scenario for security analysts, and for democracy—it's the briefcase nuke in downtown Manhattan, or the human-transmissible bird flu strain in the international terminal of LAX.

Because undetectable wholesale election fraud is the holy grail of anyone who wants to steal an election, I'll spend the rest of this article discussing it in some detail. Along the way, you'll also see that most of the attacks I'll cover can also be carried out on the retail level, as well.

Narrowing the focus: the Diebold AccuVote TS

Even after the passage of the Help America Vote Act (HAVA) in 2002, national election standards at all levels of the electoral process—site security, machine security, election procedures, auditability requirements, dispute resolution, etc.—are either extremely weak or, in many cases, simply ignored by states and counties. Because of the extraordinary variability of voting technologies and procedures from state to state, the entire country presents a morass of special cases to the writer who would lay out a generally applicable scenario of electronic election theft.

Because the technologies, techniques, and procedures at issue vary so widely, it's necessary for me to narrow the focus of the present discussion to one particular DRE voting machine: the Diebold AccuVote TS.

The Diebold AccuVote TS is one of the most popular DRE voting machines currently in use. Georgia and Maryland have both standardized on this model across the state, and Diebold claims that over 130,000 of its AccuVote TS and TSx (an updated model) machines are now in use across America.

Processor	118 MHz Hitachi SH3
Storage	16MB RAM, 32MB on-board Flash, 128K EPROM
I/O	Keyboard, modem (PCMCIA), IrDA, headphone jack
Firmware	Custom Diebold firmware
Operating system	Windows CE 3.0
Application software	Custom Diebold system software
Display	Touchscreen, thermal roll printer (for printing a zero tape and final vote tallies)

The AccuVote TS is also the DRE that has been subjected to the most scrutiny by the infosec community, mainly because its source code has been widely available on the Internet. Much of what I'll say about the AccuVote will apply to other DRE systems as well. Some specific vulnerabilities, like the unencrypted ballot definition file described later, are probably peculiar to the AccuVote, but many of the overall types of attacks enumerated here apply to other DREs. (It's hard to say which other other DREs are vulnerable to which attacks, because we don't have source code for the others so it's harder to know how secure they are.)

Casting (and cracking) a vote on the Diebold AccuVote TS

In a previous section, we went over the basics of voting on a DRE. Now let's step back a bit and look at a picture of the entire voting process using an AccuVote.

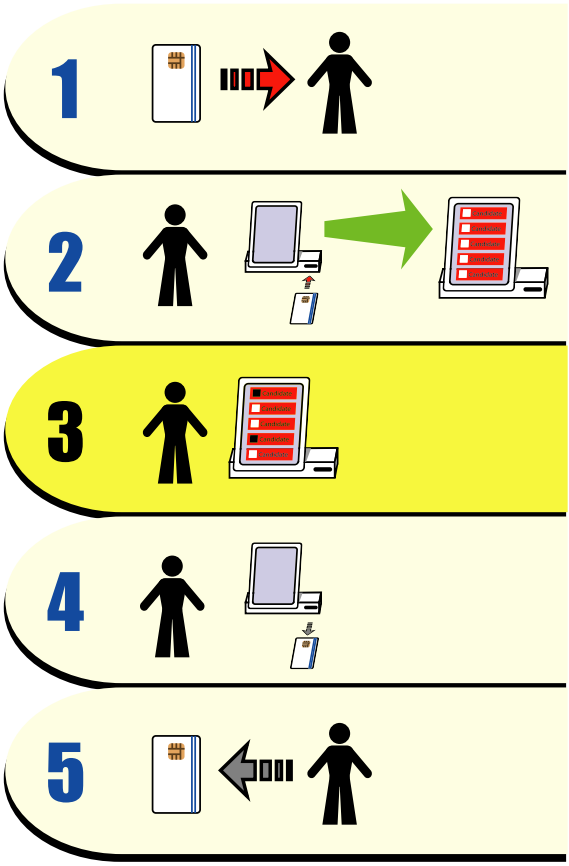


Figure 3: Electronic voting using a Diebold AccuVote TS

Here are the steps described in detail:

1. After showing proper identification, the voter is issued an activated smart card. This card enables the voter to vote one ballot and one ballot only.
2. The voter inserts the smart card into the machine. Once inserted, the smart card tells the AccuVote which races the voter is authorized to vote in. The AccuVote then loads the *ballot definition file* (BDF) that's appropriate for that voter. The AccuVote's internal software uses the BDF to display the ballot on the touchscreen.
3. The voter votes by touching his selections on the screen. Once the electronic ballot is complete, the machine asks the voter to verify his selections before recording them directly onto an internal storage device. The AccuVote's internal storage device is a PCMCIA Flash memory card.
4. The voter removes the smartcard, which is now deactivated and cannot be used again until it is reactivated.
5. The voter returns the smartcard to the poll worker, who then reactivates it for issuing to another voter.

The voting process described here is vulnerable to multiple types of retail fraud at almost every point. Because the focus of this article is on wholesale fraud, I'm only

going to briefly outline a few of the retail fraud mechanisms, just to give you a taste for Diebold's overall approach to security:

- The Ohio Compuware report describes how to turn a voter card into a supervisor card, which can then be used to cast multiple votes, delete votes, or shut down the machine, using a PDA with a smartcard attachment.
- In order to use a supervisor card to access the AccuVote, you must first enter a four-digit PIN. In version of the machine that was in use as late as 2003, the exact same supervisor PIN was hard-coded into every single AccuVote TS shipped nationwide. That PIN was 1111. (I am not making this up.) This is still the default PIN for these machines, although the county can change it on a machine-by-machine basis if they have the manpower and the time.
- All of the AccuVotes have the same lock securing the PCMCIA slot that contains the Flash card with all the votes on it. When I say the "same" lock, I mean the exact same key opens all of the machines. But even if you don't have one of the tens of thousands of copies of this key that are floating around, the lock can be picked by an amateur in under 10 seconds. The Princeton video has a nice demo of this. Once you have access to the PCMCIA slot, you can do all kinds of great stuff, like upload vote-stealing software (a simple reboot will cause the machine to load software from whatever you've put in the PCMCIA slot), crash the system, delete all the votes on the machine, etc.
- Some localities have taken to securing the PCMCIA slot with security tape or plastic ties. The idea here is that a cut tie or torn tape will invalidate the results of that machine, because poll workers can't guarantee that it wasn't compromised. There are two things wrong with this scheme:

    1. If you want to invalidate all the results stored in machines in a precinct that favors your opponent, just cut the tape or the ties on those machines. If the election supervisor sticks to the rules, then he or she will be forced to throw out all of those votes.
    2. According to author, security researcher, and Maryland election judge Avi Rubin, one would almost have to have a CIA background to be able to tell if the security tape applied to the AccuVotes in the Maryland primary had been removed and reapplied.

I won't rehearse the rest of the long list of retail fraud opportunities made available by the AccuVote TS. Some searching will turn up dozens of reports and thousands of web pages with as much detail as you can stand on how to create mischief with these machines in a polling booth. Now it's time to move on to the good stuff: undetectable wholesale fraud.

Wholesale fraud on the AccuVote TS

In order to understand how we can commit wholesale fraud on the AccuVote TS, we first need to know a bit more about how the system is structured. In particular, we have to take a closer look at the unit's software, and how it records votes.

Computer scientists often speak of the multiple levels of software that make up a system as a "software stack." Each layer in the stack supports the layer above it, and malicious code in a low-level layer can affect all of the layers above.

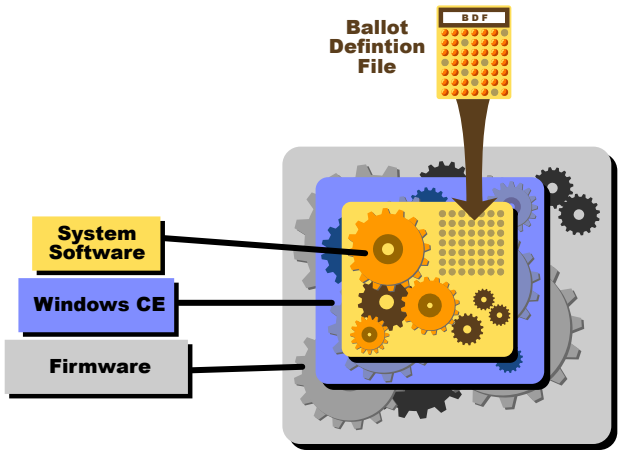


Figure 4: The Diebold AccuVote TS software stack

As you can see from Figure 4, the AccuVote's software stack consists of three primary layers. At the lowest level, closest to the hardware, sits the firmware layer. The AccuVote's firmware is the first program to be loaded into memory when the machine boots, and it takes care of loading the next layer of the stack, which is the operating system.

***Note**: Because all of a DRE's software loads from a pool of internal Flash memory, DRE vendors tend to refer to every piece of software in the system as "firmware." In this article, I'll stick to the standard firmware/OS/application distinction, just to avoid confusion.*

The AccuVote's operating system is a custom version of Windows CE. Diebold licenses Windows CE from Microsoft and modifies it to fit the AccuVote. (For the uninitiated, the operating system is really a collection of different software libraries that handles all of the low-level tasks in the system, like reading and writing to the internal storage device, displaying things like windows and checkboxes on the touchscreen, managing files and applications, and so on.)

When Windows CE boots on the AccuVote, it loads the main system software application that actually handles the ballot display and voting process. The system software selects the proper ballot definition file to present to the voter, and it then uses that file to record the voter's selections on the Flash memory card.

So with this concept of a software stack in mind, let's expand step 3 from Figure 2 to see exactly how the AccuVote records the voter's touch-screen selections.

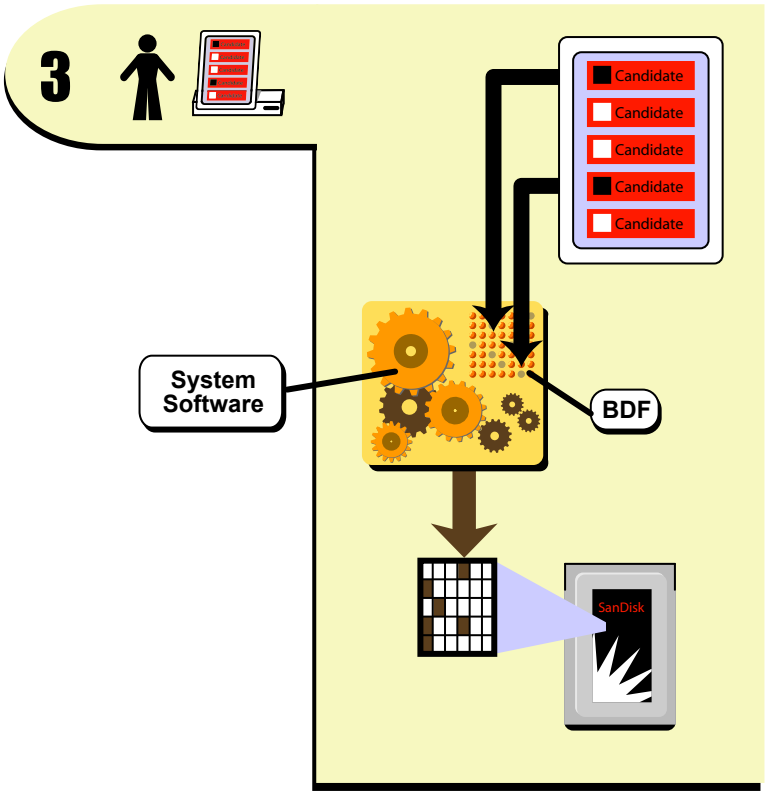


Figure 5: Casting a vote on the Diebold AccuVote TS

As you can see in Figure 5, the voter's selections are read from the touch screen by the AccuVote's internal system software. The system software uses the BDF to translate the selections into a format that can be written to the internal storage card, where they're stored along with all of the other votes cast on that machine.

If you were going to steal an election with an AccuVote, one of the best and easiest methods is to manipulate the BDF. On the AccuVote, the BDF is completely unencrypted, so it just sits there in the machine's memory open to all comers. Malicious software embedded in any layer of the software stack can easily get at the BDF and alter it so that selections made for one candidate are recorded on the machine's memory card for another candidate. If the software is programmed to remove itself after the election, then there would be absolutely no way for anyone to know that the results are fraudulent.

Of course, an attacker with access to any or all of the layers of the software stack can do more than just manipulate the BDF so that votes are misrecorded in real-time. He could conceivably ignore the BDF entirely and just change the machine's vote totals directly on the memory card, so that they produce a desired outcome. Indeed, just as is the case with a regular personal computer, the possibilities for a malicious Trojan to make mischief on the DRE is limited only by the skill and imagination of the attacker.

Ed Felten's team at Princeton was able to quickly upload a vote-stealing Trojan to the AccuVote via the PCMCIA slot in less time than it would take many people to complete an electronic ballot. Furthermore, they also created a viral version of the Trojan that could infect any card inserted into the PCMCIA slot with vote-stealing software that would then infect any machine into which the tainted card was

inserted. The newly infected machines would in turn infect other cards, which would infect other machines, and so on. In this way, the vote stealing "Princeton virus" could travel across an entire precinct or county, given enough time.

The viral nature of the Princeton attack is one way to commit wholesale undetectable vote fraud, but there are others that are even more efficient and require no physical access to a machine at any point. Specifically, if any one of the institutions responsible for loading software onto the AccuVote (or any other DRE for that matter) has been compromised, either by an internal mole or an outside cracker who has hacked into the company's internal network, then something like the Princeton virus could be planted in the firmware, operating system, or system software build that goes on machines across an entire county or state.

In other words, you know how Apple just accidentally shipped a few thousand iPods with a Windows virus embedded in them? If you replace "Apple" with "Diebold" and "iPod" with "AccuVote," then you've got a recipe for wholesale election theft.

Think about that for a moment, and let it sink in. To have confidence in the results of an election using DREs, you no longer have to put your trust solely in the security practices at the Board of Elections. Now, you have to have confidence in the security of the DRE vendor's corporate networks, and in their human resources departments, and in the security practices and personnel of anyone else who touches the software that goes into a DRE (i.e. a third-party software vendor).

To give you some perspective on the level of security at voting machine companies, there have been actual incidents that involve intruders breaking into the internal networks of three DRE vendors and gaining access to sensitive information:

1. A hacker penetrated VoteHere's intranet in 2003.
2. Diebold was also the victim of a hacker in 2003, in a highly publicized intrustion in which thousands of internal company emails were stolen and made public.
3. ES&S was burglarized in 2003, and sensitive information, including voting software, was stolen. The company didn't notify the public until three years later.

Figure 6 gives you a visual breakdown of the three main institutions that contribute layers to the AccuVote's software stack: the county Board of Elections, Diebold, and Microsoft. Again, one well-placed bad apple in any one of those institutions, or an unauthorized intruder with access to the right network, could steal a state-wide election in Georgia, Maryland, or any other county or precinct that relies on the AccuVote TS.

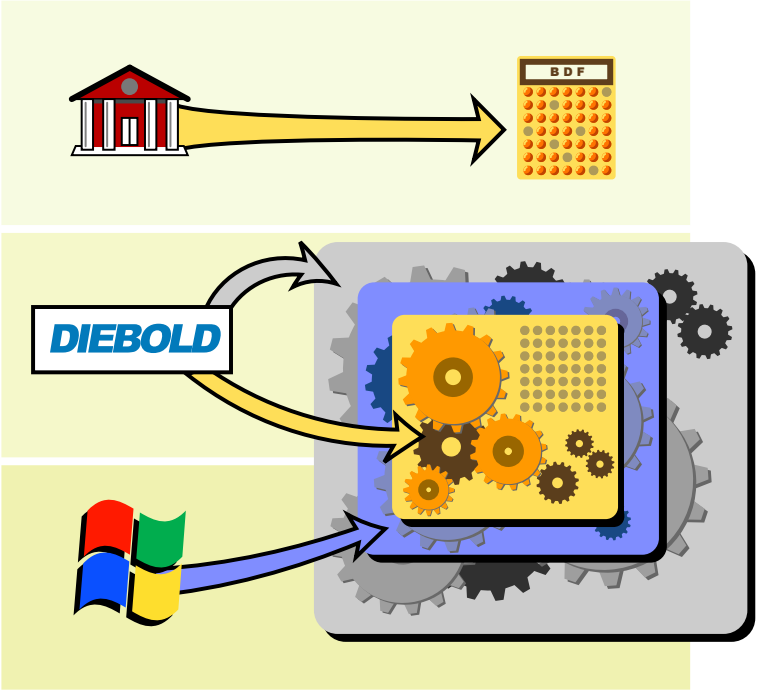


Figure 6: The Diebold AccuVote TS software stack

In some cases, the BOE isn't actually involved in creating the ballot definition file. The county's election workforce is so understaffed and starved for volunteers, and the rollout of DREs before an election is so rushed, that some counties and just let Diebold come in and handle the entire election—BDF creation, certification, logic and accuracy testing, set-up, tear-down, the works. The whole election is just handed over the private sector to run, with the county providing practically no oversight because they don't even really know how to use the systems without hand-holding.

Logic and accuracy testing

One of the last lines of defense against the kinds of intrusions described here is the logic and accuracy (L&A) test. The idea behind the L&A is that voting machines are put through a mock election by county officials, and their outputs are compared to their inputs to confirm that the machines are faithfully recording the totals.

There are a few problems with the L&A as a barrier against election fraud. First, the Princeton virus can tell when the machine is doing an self-run L&A test, and it will produce correct results under those conditions. Second, it's not at all difficult to imagine how a Trojan programmer would detect that an L&A test is being carried out: check the system clock to see if the voting is taking place on election day, or on some other day; see if the number of votes cast is less than the expected number; see if the polling lasts for a shorter period of time than expected; and so on.

Finally, each L&A test takes time, which is why it's impossible to fully test every single DRE before an election. If you could ensure that all of the software on a pool of DREs is exactly the same, then you could fully test one DRE in a realistic mock election and be done with it. Such a testing protocol would catch any Trojan embedded in the software stack that was written by an author who's not creative enough to fool a really thorough L&A test. But even the most rigorous and realistic L&A test couldn't thwart a "knock attack."

Briefly, a "knock attack" is where the Trojan doesn't wake up and do its business until it receives a signal of some sort from the attacker. For networked machines, this could be something as simple as a scan on a certain port range. For non-networked touchscreen machines, Avi Rubin has suggested that an attacker could touch the screen in certain place, or make a sequence of specific touches (e.g. top left, top right, top left). Or, an attacker could send a signal to the AccuVote's built-in IrDA port with a handheld remote (if there's an IR sensor actually installed and accessible). There are a number of possibilities here, but you get the idea.

Realistically, the L&A is just one of a series of tests that should take place at every step of the voting machine procurement, deployment, and election process. The machines should be audited independently and tested by the government before they're purchased by the state or county; they should be tested on delivery; they should be tested prior to polling; and a random sample should be tested during polling.

Fundamentally, however, it doesn't matter how thoroughly you test a paperless DRE before, during, or after an election. A determined cracker can always find a way to compromise the system in an undetectable way. The only real protection against wholesale election fraud is genuine auditability, and that's a feature that paperless DREs lack by design.

So far in this article, I've covered two of the three bullet points that I listed for undetectable wholesale fraud methods: altering the vote recording process, and altering the record of votes. Now let's look at the remaining fraud method: altering the tabulation process.

(Mis)counting the vote

You might have a hard time imagining that a company like Diebold could ever be compromised from within or without by someone who would wand to steal an election by embedding a Trojan in the AccuVote's software stack. Or, alternately, you might have faith that the testing and voting machine certification process in your state is thorough enough to catch even the most cleverly hidden Trojan. Even so, you still shouldn't be complacent about DREs, because there are other moments in the life-cycle of an electronic vote where that vote can be altered.

Figure 7 shows the process by which votes are collected and tabulated.

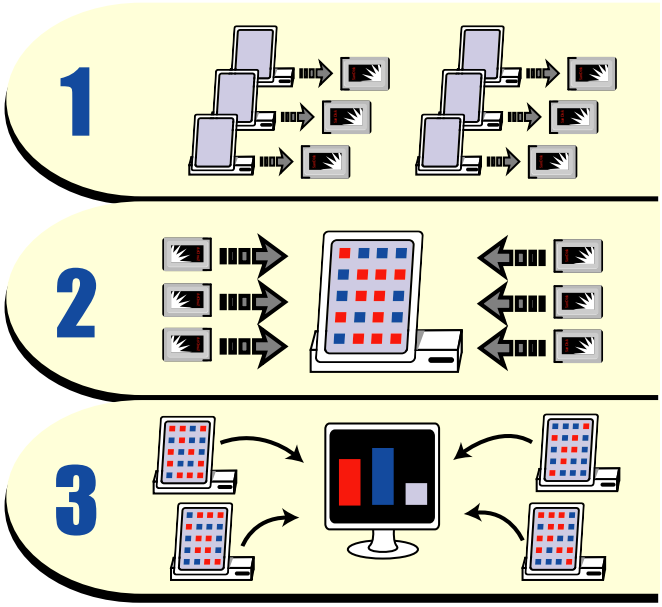


Figure 7: Tabulating the vote with Diebold hardware and software

The steps in the process are as follows:

1. First, the memory cards are removed from all of the machines in a precinct.
2. One of the machines is designated as an *accumulator*, which means that it's that machine's job to read all of the memory cards, one-by-one, and compile all of their votes into one master list. So all of the removed memory cards are inserted in the accumulator, one at a time, to have their contents uploaded.
3. All of the accumulator machines in all of the precincts dial into one or more PC servers running Microsoft Windows and Diebold's General Election Management Software (GEMS). Once the accumulators connect to the GEMS server, their vote totals are downloaded and compiled, and an official tally is made.

Note that DREs from some vendors are made to be networked together throughout a precinct via Ethernet or wireless. In such a configuration the accumulator machine can download all of the votes from the other machines over the network, so no memory cards need to be removed from one machine and reinserted into another.

Those of you who've followed the article thus far and who have any knowledge of information security will immediately spot the vulnerabilities in the process outlined above. Let me run through a few of the opportunities for wholesale fraud that this scheme provides.

First, if the accumulator DRE happens to be running something like the Princeton virus, then it's game over. That one machine can flip the totals on every card that's inserted into it, and there will be no way to detect that any fraud has occurred. If this were happen, all of the results from an entire precinct would be tainted because of one compromised DRE.

If all of the machines in the precinct are networked (God forbid!), then stealing an entire precinct's votes gets even easier. A single compromised machine could infect

the accumulator and every other machine on the network, tainting all of the results for that precinct. And if those machines are networked wirelessly(!!), then a fraudster with a laptop and a wireless card in a car outside the precinct building could conceivably have his way with all of the votes in the building.

Cracking the central tabulation (GEMS) server

The GEMs server deserves special attention as a weak point in the design of the overall system. This server is a typical PC with a typical PC software stack. In fact, I could conceivably reuse my depiction of the AccuVote TS software stack in Figure 6 by replacing "Windows CE" with "Windows XP," "System Software" with "GEMS," and "BDF" with "GEMS database."

The GEMS database stores all of the votes collected from precinct accumulators, and it's used to do the vote tabulation for a county. Because it's so sensitive, you might think it would be tightly secured. But you'd be wrong.

The GEMS database is a vanilla, unencrypted Access database that anyone with a copy of Microsoft Access can edit. So if you have physical access to the GEMS server's filesystem (either locally or remotely), then it's not too hard to just go in and have your way with the vote totals. If Access isn't installed, just install it from a CD-ROM, or connect remotely from a laptop and edit the database that way.

Or, if you want to filch the database, upload vote-stealing software, or do something else evil, you could always carry along a USB drive in your pocket.

Many GEMS servers are connected to a modem bank, so that the accumulators can dial in over the phone lines and upload votes. One team of security consultants hired by the state of Maryland found the GEMS bank by wardialing, discovered that it was running an unpatched version of Windows, cracked the server, and stole the mock election. This great Daily Show segment, in which one of the team members describes the attack, states that they did this in under five minutes.

If the GEMS server is somehow connected to the Internet, and some of them are (in spite of Diebold's strong recommendation that they not be), then any one of a billion script kiddies who can crack a Windows box can have a field day with the election...

I could go on here with the hypotheticals, but let's take a look at how this is alleged to have played out in the real world, this past August in Shelby County, Tennessee:

> Evidence from election official declarations and discovery documents obtained in litigation over a recent election using Diebold machines reveals that:
>
> - Illegal and uncertified Lexar Jump Drive software was loaded onto the Diebold GEMS central tabulator, enabling secretive data transfer on small USB "key chain" memory devices. This blocked election transparency and raises questions as to whether hidden vote manipulation may have taken place.
> - Other uncertified software of various kinds was loaded onto the system and, according to the event logs examined, was used. This opened the door for hand-

editing of both vote totals and the reporting of election results.

- Evidence of actual attempts to manipulate election reporting results exists. The evidence available wouldn't record successful manipulation, only attempted manipulation, due to software failure. The logs show repeated failed attempts to use an HTML editor.
- According to Shelby County elections officials, they opened the central vote totals repository to widespread network connections. The dispersed nature of access to the central tabulator would prevent finding the perpetrators, even if documentation of manipulation could be achieved -- a difficult feat, since the type of hacking enabled by the GEMS program tends to erase evidence.

  In an on-site inspection of the network connections conducted by Jim March, elections department lead computer operator Dennis Boyce pointed to a location on a network interconnection plug panel where the Diebold-supplied GEMS central tabulator is plugged in. No extra security such as a router or firewall was present at the interconnection. This appears to open up access by anybody in county government to the central tabulator.
- At the same on-site inspection, the Diebold-supplied GEMS backup central tabulator had more uncertified software than could be quickly documented Ð but observers did spot Symantec's PC Anywhere utility. This program would allow opening the machine to outside remote control - the PC Anywhere program allows a remote computer across a dial-up or networked connection to see the screen of the "zombied" computer and operate it's keyboard and mouse. To call this a security breach is an understatement.
- At the primary GEMS central tabulator station, all of MS-Office 2000 Professional was loaded and working. According to Windows, MS-Access was a frequently used program, the only component of the overall MS-Office suite that was so identified.

Note that I haven't done any journalistic due diligence on this particular report, so I'm obliged not to vouch for its absolute veracity. But my point in reproducing it is that every one of these items is 100 percent plausible, so this incident report paints an extremely realistic portrait of how the GEMS server could be compromised to steal an election.

Finally, before I leave this topic, I want to raise the possibility that a DRE manufacturer could include an undocumented back door in the GEMS server that would leave the machine open to manipulation and fraud. Of course, it may be more than just a possibility. Such a back door has allegedly already been found, as referenced in this CERT bulletin. However, the details here are sketchy, and one researcher that I've talked to says the credibility of this report is suspect. Also, I'm going to give Diebold the benefit of the doubt and assume that this back door (if it exists) was put there for maintennance and/or testing reasons, and that it was never intended to be enabled on a production build of the software.

Spoofing the GEMS server

Physical or remote access to the GEMS server gets you the keys to the electoral kingdom, but those aren't the ways to exploit the GEMS server to rig an election. To understand another good way to manipulate this system, we have to return to our friend the ballot definition file (BDF).

One of the most shocking revelations that the Johns Hopkins team uncovered in their security analysis of the AccuVote is that the BDF contains all of the information necessary to connect to and upload votes to the GEMS server. From p.22 of Avi Rubin's new book, *Brave New Ballot*:

> We found that in addition to this basic data, the ballot definition file contained more sensitive, security-critical information, including the voting terminal's voting center identification number, the dial-in numbers for the end-of-the-day tally reporting, the network address of the back-end processing server, and a username and password. It was like finding somebody's wallet: in this file you'd have everything needed to impersonate the voting machine to the board of elections servers. Since there was no cryptographic authentication between the voting machines and the tallying servers, someone with a laptop and the information from the ballot definition file could dial into the board of elections computers from *anywhere* and send in fake vote tallies.

Rubin goes on to allege that after the release of the Hopkins report, Diebold claimed that they fixed this problem. Then a subsequent report showed that, no, they hadn't fixed it. So in response to the new report Diebold claimed to have fixed it again. Who knows if it ever truly got fixed—the Diebold source is closed and proprietary, so we have to continue taking their word for it.

The bad apple chart

The term "black box voting" is commonly used by e-voting activists to describe the non-transparent way in which elections are carried out using DREs, with the idea being that the DRE is a "black box" that tallies votes in an invisible, proprietary, and potentially suspect manner. For my part I think the term "black box" best describes not the DRE, but the DRE manufacturer. The entire voting machine company—its corporate network, its management, its staff, its internal policies and procedures—is a giant black box that we, the voters, must trust is free of malicious influences from within and without.

So if you learn one thing from this article, I hope it's this: DRE's multiply tremendously the sheer number of institutions and people that you have to trust in order to have confidence in an election's results. In this last part of the article, I'd like to give you a feel for who you're relying on when you walk into a polling booth this November and make a touchscreen selection for your candidate of choice.

Take a look at Figure 8, which is diagram of inputs and outputs from a generic DRE. This is my own version of a diagram that appears in the Ohio Compuware report.

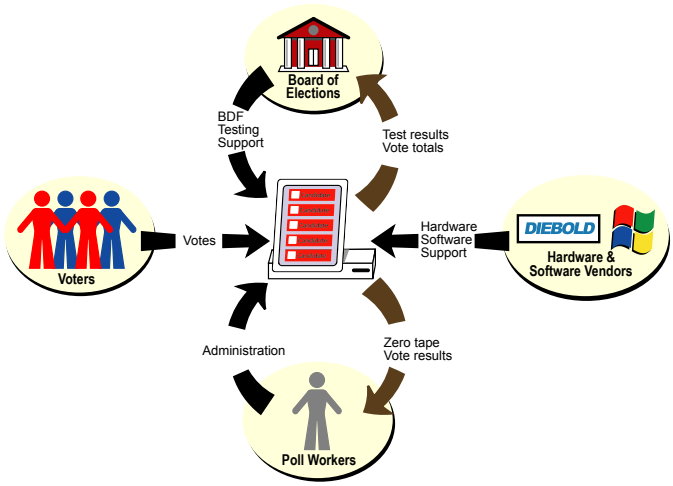


Figure 8: Who interacts with a DRE

Throughout the course of this article, I've outlined some ways in which a single bad apple in any of these groups could compromise election results. Now I'll sum up that analysis in what I'll call a bad apple chart (really more of a diagram than a chart), shown in Figure 9:

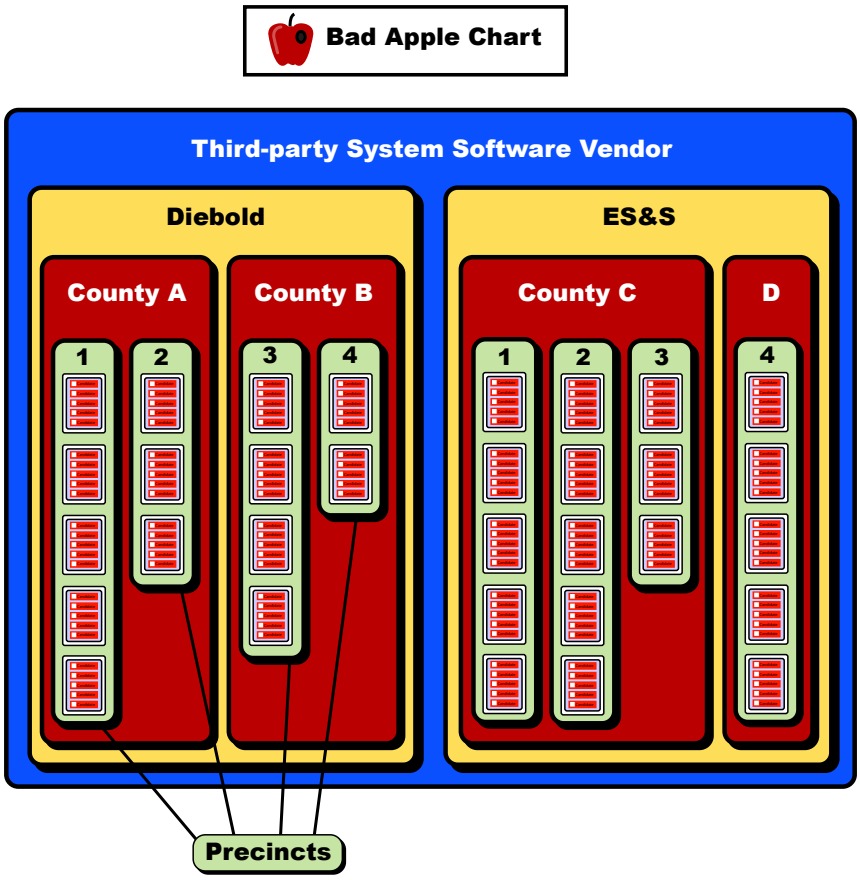


Figure 9: The Bad Apple Chart

The basic idea behind the chart is that you can place a bad apple in any one of the boxes, and any number of the voting machines within that region could be compromised. The "third party system software" referenced in the outermost box could be any third party software, like an multiple vendor (here Diebold and ES&S, for example). Finally, note that the counties and precincts are sized differently, just to show some variation.

If you wanted to steal an election, the best place to drop a bad apple would be at a popular operating system vendor. E-voting expert Douglas Jones has proposed the following such scenario, merely as an example to show what's possible:

> In the next version of their window manager, a major vendor includes a little bit of code as part of the "open new window on screen" mechanism. If today is the first Tuesday in November of an even numbered year, this code checks the contents of the window. If the window contains the strings "Straight Party", "Democrat", "Republican", "Socialist", and "Reform", and if the window contains a "radio button" widget, allowing the selection of one out of n alternatives, the software would, one time out of ten, exchange the words "Republican" and "Reform".
>
> What does this little bit of code do? On election day, and on no other day, it throws ten percent of the straight party Republican vote to a large third party that is known to attract many Republican-leaning voters. In closely contested Democratic-Republican contests, this could easily swing the outcome to favor the Democrats, and on a national scale, it could easily provide the winning margin for control of Congress or the White House...
>
> This kind of attack does not require either massive conspiracy or corporate approval or cooperation! So long as a single programmer can covertly incorporate a few lines of simple code into a component that he or she knows will end up in a large fraction of all voting machines, and so long as that code is not subject to exhaustive inspection, the system is vulnerable! Someone intent on fixing an election does not need to buy the support of the company, they only need to buy the support of one programmer with access to a key component!

If you don't think that it's possible someone to buy off, say, a Microsoft employee with access to the right window manager libraries, or you think that Microsoft would eventually catch the crack with a source code review (even in spite of clever obfuscation on the part of the mole), then you'll be heartened to know that Jones has confirmed my suspicion that a virus could easily make the modification described above... or, it could make some other, equally clever modification that no one has thought of yet. All that's needed is to get the virus onto a machine at a DRE vendor that houses builds of one or more layers of their DRE's software stack, and you have the capability to do undetectable wholesale fraud.

This last point brings me to next region into which a bad apple could be profitably inserted: the DRE vendor (or the vendor's network). There's no need to say much more about this, though, because most of the article has been taken up with this type of scenario. Bad apples in this area can commit undetectable wholesale fraud.

At the county level, a worker at the BOE has many, many opportunities to commit wholesale fraud by exploiting her regular access to the "machinery of democracy" at all points in the election process to upload vote-stealing software onto a DRE, and accumulator, a central tabulation server, or all three.

Finally, at the precinct level, it's possible for a single bad apple (a poll worker, or even a voter) to commit any number of bad acts: disenfranchisement of a precinct by means of vandalism, multiple voting, deleting votes, uploading vote-stealing software, etc.

Wholesale fraud at the precinct level

You might think that you'd have to commit an infeasibly large number of acts of precinct-level fraud to steal an entire election, but you'd be wrong. Gerrymandered voting districts, in which whole precincts lean heavily in one direction or another, make disenfranchisement attacks on precincts a highly effective form of election fraud.

For attacks like this, urban voters are especially vulnerable, because they have a higher number of populous precincts clustered together in a smaller geographic area. It's much easier to use vandalism (disguised as machine malfunction) to disenfranchise multiple urban precincts on election day than it is go all over the countryside and suburbs in a state like Ohio to break voting machines that are scattered in isolated elementary schools.

Finally, it's extremely important to note that, in the absence of a meaningful audit trail, like that provided by voter-verified paper receipts, **it is virtually impossible to tell machine malfunction from deliberate vandalism**. Pioneering election security researcher Rebecca Mercuri has told me that she's actually much more concerned about "disenfranchisement of voters due to the strategic denial-of-service that currently masquerades as malfunctions," than she is about "manipulation of election equipment and data files in order to alter election outcomes, although both remain problematic."

When you have a rash of voting machines that have their memories wiped, their votes erased, or their number of votes mysteriously inflated; when you have reports of machines that crash or refuse to respond; when many machines record a vote for the wrong candidate—all of this could just as plausibly be construed as evidence of fraud as it could be of spontaneous malfunction, because there's simply *no way* to tell the difference in most cases.

Conclusions: take-home points and parting thoughts

The picture that I've painted here about the state of the American electoral system is bleak and depressing. Even more depressing is the fact that *absolutely nothing* can be done to address these vulnerabilities in any substantial way before the November midterm elections. Really, the only thing that citizens can do for the midterms is get involved by volunteering at their local precinct and keeping their eyes and ears open. Watch everything, and record everything where possible.

Right now, the only thing standing in the way of the kind of wholesale undetectable election theft that this article has outlined is the possibility that DREs were forced onto the public too rapidly for election thieves to really learn to exploit them on this cycle. There's always a gap between when a security vulnerability is exposed and

when it's exploited, so let's all hope and pray that November 7 falls within that time window.

In the medium- and long-term, it is just as much of a certainty that many of these vulnerabilities will be exploited as it is that, say, major new Windows security vulnerabilities will be exploited. Indeed, the stakes in stealing an election are much, much higher than they are in the kind of petty hacking that produces today's thriving ecosystem of PC viruses and trojans. I've outlined the way (already widely known) in this article, and I don't doubt that someone, somewhere, has the will to match that way. Unless security practices and electoral procedures are upgraded and standardized across the country, and unless meaningful auditability is mandated (preferably a voter-verified paper trail) nationwide, then the probability of a large-scale election theft taking place approaches certainty the longer we remain vulnerable.

In conclusion, let me summarize what I hope you'll take home with you after reading this article and thinking about its contents:

- Bits and bytes are made to be manipulated; by turning votes into bits and bytes, we've made them orders of magnitude easier to manipulate during and after an election.
- By rushing to merge our nation's election infrastructure with our computing infrastructure, we have prematurely brought the fairly old and well-understood field of election security under the rubric of the new, rapidly evolving field of information security.
- In order to have confidence in the results of a paperless DRE-based election, you must first have confidence in the personnel and security practices at these institutions: the board of elections, the DRE vendor, and third-party software vendor whose product is used on the DRE.
- In the absence of the ability to conduct a meaningful audit, there is *no discernable difference* between DRE malfunction and deliberate tampering (either for the purpose of disenfranchisement or altering the vote record).

Finally, it's worth reiterating that optical scan machines are vulnerable to many of the same exploits as the DREs on which this article focuses. Optical scan machines do leave a paper audit trail, but that trail is worthless in a state (like Florida) where manual audits of optical scan ballots are not undertaken to clear up questions about the unexpected returns from certain precincts. I've been told that such audits are now prohibited in Florida by law in the wake of the 2000 voting scandal.

Postscript

In researching this article, I talked on- and off-the-record with a number of prominent experts in the electronic voting field. The following e-mail response from computer scientist and e-voting/security expert Peter Neumann sums up the present state of chaos heading into the November midterm election, and it also communicates some of the frustration (and fear) that I heard echoed in the responses of the other researchers whom I questioned.

> The problem is much deeper than most people realize. The standards are extremely weak (1990 and 2002 both), and VOLUNTARY. The systems are built to minimum standards rather than attempting to be meaningfully secure. The evaluations are commissioned and paid for by the vendors, and are proprietary. The entire voting process consists of weak links—registration, voter disenfranchisement, voter authentication, vote casting, vote recording, vote processing, resolution of disputes (which is essentially nonexistent in the unauditable paperless DREs), lack of audit trails, and so on. You cannot begin to enumerate the badness of the present situation.

Paradoxically, the media blizzard of disparate facts, figures, vulnerabilities, acronyms, and bad news from a huge list of states, counties, and precincts, is in large measure responsible for the current lack of an all-out panic among the public and political classes as we head into the November mid-terms. This steadily roiling storm of e-voting negativity has resulted in a general uneasiness with DREs among the public and the media, but threat feels diffuse and vague precisely because there are just so many things that could go wrong in so many places.

To get a sense of the problems that security researchers have in boiling all this bad news down into a single threat scenario that's vivid enough to spur the public to action, just imagine yourself travelling back in time to 1989 to testify before Congress about "the coming plague of identity theft." Or how about, "the rising terrorist threat from Islamic fundamentalism."

My own personal fear is that, by the time a whistleblower comes forth with an indisputable smoking gun—hard evidence that a large election has been stolen electronically—we will have lost control of our electoral process to the point where we will be powerless to enact meaningful change. The clock is ticking on this issue, because a party that can use these techniques to gain control of the government can also use them to maintain control in perpetuity.

Bibliography and further reading

Sites

- Doug Jones, Voting and Elections
- Princeton University, Center for Information Technology Policy
- Peter G. Neumann's Site
- Rebecca Mercuri's Electronic Voting page
- Blackboxvoting.com
- Bev Harris, Blackboxvoting.org
- Verified Voting
- Voters Unite
- Election Integrity
- Open Voting Consortium

Blogs

- Avi Rubin's Blog
- E-Voting Experts
- Freedom to Tinker
- The Brad Blog

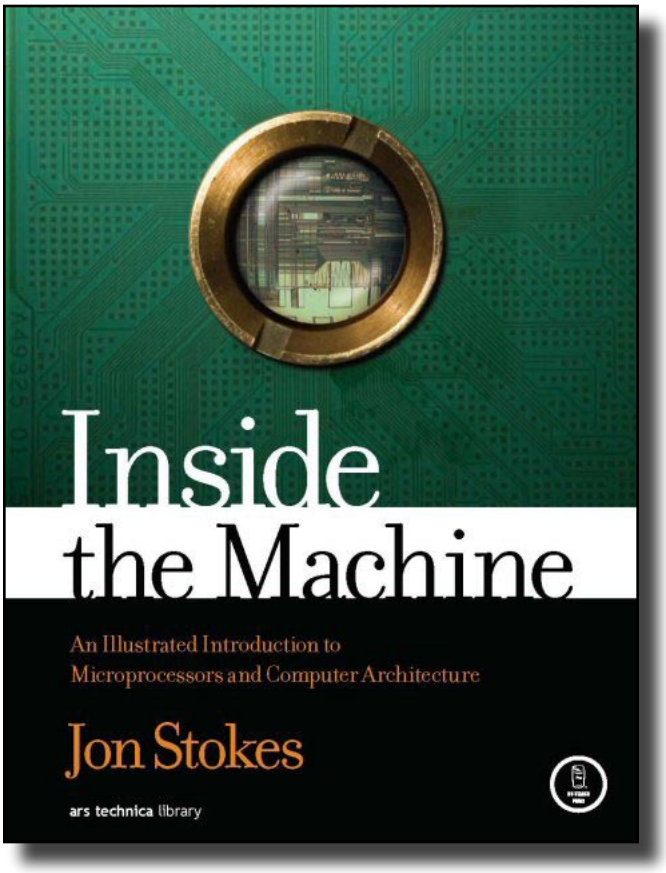
Reports

- Maryland SAIC Report
- Maryland RABA Technologies Report
- Ohio Compuware Report
- Avi Rubin et al, Analysis of an Electronic Voting System
- Election Data Services, Inc., 2006 Voting Equipment Study
- Common Cause, Election Reform, Malfunction and Malfeasance: A Report on the Electronic Voting Machine Debacle
- Harri Hursti, Diebold TSx Evaluation, SECURITY ALERT: May 11, 2006: Critical Security Issues with Diebold TSx
- Ariel J. Feldman , J. Alex Halderman , and Edward W. Felten, Security Analysis of the Diebold AccuVote-TS Voting Machine

Books and articles

- Avi Rubin, Brave New Ballot
- David Wagner, Testimony of Dr. David Wagner before House Committee
- Rebecca Mercuri, DRE Voting -- Designed for Failure
- Rebecca Mercuri, Vincent J. Lipsio, and Beth Feehan, COTS and Other Electronic Voting Backdoors
- David L. Dill, Bruce Schneier, and Barbara Simons, Voting and Technology: Who Gets to Count Your Vote?
- Kim Zetter, How E-Voting Threatens Democracy
- Another source code leak for Diebold
- Diebold voting machines hacked in Florida
- Diebold voting machine failures strike again in Alaska
- Researchers find (more) severe flaws in Diebold voting machines

- Don't touch the Diebold touchscreen machines
- California to sue Diebold over voting machine-related fraud
- Diebold loses legal case, certified anyway
- Diebold withdraws from North Carolina
- Electronic voting: isn't it about time we had a collective fit of national hysteria?



Inside the Machine

An Illustrated Introduction to
Microprocessors and Computer Architecture

by Jon "Hannibal" Stokes

$49.95 COVER PRICE
HARDCOVER, 320 PP.
ISBN 1-59327-104-2

**Coming Soon!
December 2006**

Get the book from ArsTechnica (arstechnica.com)

Also available at fine bookstores everywhere, including Barnes & Noble, Borders, and Amazon.com, and directly from No Starch Press (www.nostarch.com)

ars technica NO STARCH PRESS

A Look Inside the Silicon Heart of Modern Computing

Computers perform countless tasks ranging from the business critical to the recreational, but regardless of how differently they may look and behave, they're all amazingly similar in basic function. Once you understand how the microprocessor—or central processing unit (CPU)—works, you'll have a firm grasp of the fundamental concepts at the heart of all modern computing.

Inside the Machine, from the co-founder of Ars Technica, explains how microprocessors operate—what they do and how they do it. The book uses analogies, full-color diagrams, and clear language to convey the ideas that form the basis of modern computing. After discussing computers in the abstract, the book examines specific microprocessors from Intel, IBM, and Motorola, from the original models up through today's leading processors. It contains the most comprehensive and up-to-date information available (online or in print) on Intel's latest processors: the Pentium M, Core, and Core 2 Duo. *Inside the Machine* also explains technology terms and concepts that readers often hear but may not fully understand, such as "pipelining," "L1 cache," "main memory," "superscalar processing," and "out-of-order execution."

Inside the Machine is perfect for students of science and engineering, IT and business professionals, and the growing community of hardware tinkerers who like to dig into the guts of their machines.

About the Author

Jon "Hannibal" Stokes is co-founder of and Senior CPU Editor of Ars Technica. He has written for a variety of publications on microprocessor architecture and the technical aspects of personal computing. Stokes holds a degree in computer engineering from Louisiana State University and two advanced degrees in the humanities from Harvard University. He is currently pursuing a Ph.D. at the University of Chicago.

NO STARCH PRESS
WHERE CONTENT AND QUALITY MATTER™

www.nostarch.com
800.420.7240
415.863.9900
info@nostarch.com