2007 USENIX/ACCURATE Electronic Voting Technology Workshop

Boston, MA
August 6, 2007

ANALYSIS I

Summarized by Kyle Derr (derrley@rice.edu)

- ***Studying the Nedap/Groenendaal ES3B Voting Computer: A Computer Security Perspective***

  Rop Gonggrijp and Willem-Jan Hengeveld, Stichting "Wij vertrouwen stemcomputers niet"

Rop Gonggrijp gave a narrative summary of his experiences in dealing with the general issue of electronic voting in The Netherlands, and he presented the findings from his security analysis of the Nedap ES3B voting computer.

Gonggrijp began his talk by explaining that while the argument for voting on paper with pencil is rather compelling in the United States, it is even more so in Holland, owing to the simple (and short) nature of the information that the average voter intends to communicate. When electronic voting machines were installed and used in Amsterdam for the first time in 2006, it was cause for significant alarm from an expert in computer security living in Amsterdam (as he was at the time). Not only were these machines most likely not secure, but they were also mostly unnecessary, as there does not exist a Netherlands analogue to what happened in the United States in Florida in the 2000 presidential election. The low-tech solution was entirely adequate.

Gonggrijp's security analysis yields two feasible attacks, which he spent the majority of his talk explaining.

The first attack demonstrates that because of the lack of physical security on the device coupled with the vintage nature of the hardware, reverse engineering, modifying, and installing malicious software is a trivially simple task. In fact, his team was able to do just this in under a month's time. The security of the physical components can easily be compromised by an intruder; the model of physical lock used to keep intruders from accessing the machine's internal components only has a single, universal key, which can be purchased for less than $5 from any one of several retailers which can be found via a simple Google search for the key's product number. The software that drives the machine is programmed on a pair of EPROMs, which are inserted into a pair of sockets on the motherboard. This property allows an attacker to swap the good pair of EPROMs with a set that contains malicious code (such as the software his team developed, which silently steals votes from randomly chosen candidates and delivers them to a preconfigured party) in less than 60 seconds.

The second attack demonstrates that voter privacy and anonymity can easily be compromised without any physical access to the device. The LCD used to display instructive information to the voter only supports mostly-ASCII text. A certain number of non-ASCII special characters can be used on the display. In the case where these non-ASCII characters are displayed, because the display controller must wait for this additional character information to be given to it from the computer, the refresh frequency of the display drops. This change in frequency is audible in some cases as far as 20 meters away with the assistance of a short-wave radio receiver. Because a popular Dutch political party has such a character in its name, a vote for this party can be detected from outside the polling place.

Gonggrijp's responses from audience questions focused mostly on rectifying confusion regarding why such an audible frequency change was being emanated from the device. In addition, Gonggrijp also forcefully noted, again, that paper ballots are more than adequate in The Netherlands. Citizens must *go for the throat* of manufacturers such as Nedap, insofar as their systems do not provide a significant advantage over paper, all things considered, accessibility gains included. When questioned specifically about the DRE's accessibility wins, Gonggrijp responded that the gain in accessibility is negligible compared to the loss in trustworthiness of the system as a whole.

- ***Security Analysis of the Diebold AccuVote-TS Voting Machine***

  Ariel J. Feldman, J. Alex Halderman, and Edward W. Felten, Princeton University

Ariel Feldman presented the results of his team's independent analysis of the Diebold AccuVote-TS. Diebold has the largest market share among US vendors: This family of Diebold machines (the AccuVote-TS and the AccuVote-TSx) currently records more than 10% of the vote nationwide. His team's contributions include a program that can silently, undetectably, and arbitrarily alter election results and a virus that can propagate this program.

Feldman first described the vote-stealing software his group designed and implemented. Because the AccuVote-TS family machines are simply general-purpose computers running a general-purpose operating system (in this case, Windows CE), Feldman and his team were able to design their software to run as a separate Windows process; patching the actual Diebold source code that runs on election day was unnecessary. The vote-stealing software simply directly edits the cast ballot data and modifies the log data such that it is consistent with the fraudulent cast ballot data. It accomplishes this by taking advantage of the fact that although this stored data is encrypted, it is encrypted with a well-known DES key (which is hard-coded in the source).

Feldman next described three different mechanisms that an attacker could exploit to install such malicious software: An attacker could boot from a malicious EPROM in an onboard socket, install a malicious boot loader by using

the software upgrade mechanism, or develop a voting-machine virus that propagates using the software upgrade vulnerability.

Whereas gaining access to the motherboard to install a malicious EPROM is a bit cumbersome, installing a memory card that contains a malicious boot loader disguised as a legitimate software upgrade is a fairly trivial task. Poll workers install and remove these memory cards as part of election day procedures; therefore all that stands between an attacker and the memory card is a locked door. Because all Diebold machines are known to be fitted with locks that can all be opened with a universal key (a key that can be obtained via several Internet retailers, including Diebold), this lock would not pose a problem to an attacker. Feldman's team was even able to clone one of these keys simply by using as a source a photo of such a key taken from the Diebold Web site. Once the malicious memory card is secured, the machine needs to be rebooted. Upon rebooting, the machine will install the software included on the memory card without asking for any sort of cryptographic guarantee that the software is from a credible source.

In addition, Feldman explained that an attacker could also exploit this software upgrade facility to create a voting-machine virus, making physical contact with every target voting machine unnecessary. The malicious boot loader installed from the attacker's memory card can clearly do much more than install the vote-stealing program: It could also disable future upgrades and copy itself to any memory card inserted into the voting machine in the future. The reason this is a credible threat can be reduced to procedure: It is common practice (indeed, Diebold-recommended practice) for a county to use several machines as accumulators in the tallying process at the end of the day. This process requires that memory cards that contain cast ballot data be inserted into the machine being used as an accumulator for the purpose of including this cast ballot data in the tally. If the accumulator machine were to be infected, it could easily infect hundreds of other machines in the course of one tally. By the same mechanism, a machine could infect the accumulator, which in turn would infect all other machines it touched throughout the course of the tally. It is reasonable to assume that a somewhat motivated attacker armed with a virus such as Feldman's could gain control of all voting machines in a county during the course of one election,in order to drastically affect the outcome of the next.

Interesting questions from the audience included one from Peter Neumann, who pressed Feldman to reveal an even more recent exploit, which allows the attacker to not have to reboot the infected machine, but simply use a buffer overflow in the running Diebold software. When asked if his virus could change the votes in real time (so as to alter printer output), Feldman responded that his team had not attempted it. When asked if a voter (as opposed to a poll worker) could easily launch the attack, he seemed skeptical, because of how obvious it would look if a voter tried to walk up to a machine and install a memory card. The session was wrapped up with Dan Wallach's wondering how Feldman responds to the common claim by election officials that election procedures defend effectively against these kinds of attacks. Feldman answered that procedures aren't always followed, and even if the ones in question were followed to the letter, they still wouldn't adequately defend against the kinds of attacks his team has developed.

■ ***An Analysis of the Hart Intercivic DAU eSlate***

Elliot Proebstel, Sean Riddle, Francis Hsu, and Justin Cummins, University of California, Davis; Freddie Oakley and Tom Stanionis, Yolo County Elections Office; Matt Bishop, University of California, Davis

Elliot Proebstel summarized four attacks and related mitigation strategies his team developed while conducting a Red Team, Black Box analysis of the eSlate DRE on behalf of Yolo county. The team found that a motivated attacker could use a covert channel to gain knowledge of a voter's choices, could cause the VVPAT printer to print duplicates easily mistaken for unique ballots, could modify vote tally data mid-day, and could use the lack of randomness in voter ID numbers to create a ballot-stuffing attack. Proebstel also offered a number of attack scenarios that could cause record inconsistencies.

The particular class of eSlate machine Proebstel's team analyzed was a Disabled Access Unit. This means it contains two features that help disabled voters. For the hearing impaired, Hart has added an audio jack near the rear of the machine, whose job is to emanate a reading of the screen content, as well as to audibly notify the voter of his or her selections. Proebstel explained that although this feature obviously makes the machine accessible to the hearing-impaired, it is also a potential source of privacy loss for the average voter. The audio can easily be captured by a hidden iPod-sized device. Because the audio must help disabled voters both through the process of entering their voter ID numbers and through the process of expressing their voting preferences, this hidden device would be able to capture enough information to link a voter to a cast ballot. Proebstel also explained that a more recent study indicates that a shortwave radio can pick up this audio, making the hidden device and physical access to the machine unnecessary. As a mitigation strategy, Proebstel offered that poll workers should be trained to detect such suspicious behavior.

Proebstel next explained a feasible ballot-stuffing attack. The machine that his team studied was connected to a thermal printer, whose purpose is to record, on paper, the voter's preference so that a manual recount is possible in case it is needed. Each printout contains a plain-text recording of the voter's preferences (intended to be verified by the voter shortly after the ballot is cast) positioned

above a barcode encoding of the same data (for ease of counting at a later date). Proebstel's team found that if communication was interrupted between the eSlate and the printer when the ballot is cast, this printer would print duplicate barcodes (without corresponding attached plain text). If the scanning of these barcodes is automated, this is a feasible ballot-stuffing attack. As a mitigation strategy, Proebstel suggested that humans sort through the printouts to ensure that each barcode is attached to corresponding plain text. An even simpler option, Proebstel suggested, is for humans to perform the entire recount.

The next attack Proebstel explained would require corruption at the poll-worker level. The eSlate machines store the effects of each cast ballot in three locations: in internal eSlate memory, in internal memory of the Judge Booth Controller, and on a removable memory card installed in the Judge Booth Controller (called a Mobile Ballot Box, or MBB). Per Hart's recommendation, Yolo county uses the MBB as the single, trusted source of election data, except in the case where a recount is necessary. Proebstel's team found that the MBB can be easily removed and modified on a simple laptop computer, then reinserted into the Judge Booth Controller, without the Judge Booth Controller noticing any inconsistency. As a mitigation strategy, Proebstel suggests strict chain-of-custody procedures be implemented and followed and that physical tamper evident seals be used on the voting machines themselves.

Proebstel's team also found that not only is the order in which voter ID numbers are chosen predictable, it is the same on all Judge Booth Controllers. When a voter walks into the polling place on election day, the voter authenticates him- or herself with election officials in some way and then is assigned a voter ID number (which is not recorded). This number is printed by the Judge Booth Controller on a small piece of paper, which is then handed to the voter. The voter then waits for a machine to become unoccupied. The voter will only enter his or her ID number into an eSlate once one becomes available. An attacker with knowledge of this predictable ID order could potentially cast ballots for other queued voters. The Disabled Access Unit input jack near the back of the eSlate only makes matters worse, as the attacker could automate much of this process. As a mitigation strategy, Proebstel suggests that only one eSlate be used per Judge Booth Controller, and only one voter ID be active at a time.

Questions from the audience included one that pressed Proebstel to compare his experience in a black box analysis of the eSlate to the one he conducted more recently, where he had access to source code. Proebstel stated that access to source made the process much easier, and he opined that now that source access is being established as precedent, future analyzers should require this. When asked if he had any advice for researchers who want to try to get election officials to work with them, rather than against them, he said he did not. In his case, Yolo county officials initiated the conversation.

DESIGN I

Summarized by Kyle Derr (derrley@rice.edu)

■ *Casting Votes in the Auditorium*

Daniel Sandler and Dan S. Wallach, Rice University

Daniel Sandler gave a narrative of his experience collecting post-election evidence on ES&S machines used in a Democratic primary election in Webb County, Texas, and how this experience motivated him to design Auditorium, a secure and distributed logging environment that makes post-election auditing easier and makes audit data much more tamper-evident and tamper-resistant. Sandler also described several potential attacks that would succeed on standard-issue logging environments but that Auditorium would prevent. The description of these attacks is omitted for brevity.

In 2006, Dan S. Wallach was hired as an expert witness by the close-second-place finisher in a Democratic primary in Webb County, Texas, after said second-place finisher received more votes on paper than he did on the ES&S DRE machine. Wallach's job was to perform a post-election audit of the records found on these ES&S machines. If enough evidence could be gathered from audit logs to prove the tallies retrieved from the ES&S machines were inaccurate, this second-place finisher would have a decent case in court. Sandler was the primary investigator in this endeavor.

Although Sandler could not find enough evidence in the logs to do his advisor's client any good, he did find many anomalies in the audit data found on the machines, such as logs starting mid-day (indicating a loss in audit data for half of the election), events taking place several days prior to the election (including 26 machines that had the same two votes cast, indicating possible inclusion of test votes in final tallies), and some machines that had no audit data at all. For the audit data that Sandler did find to be benign, he had no guarantees of the correctness or authenticity of this log data, as ES&S audit data is stored on simple, rewritable flash memory in plain text. His conclusion from this investigation was that audit data needs to be replicated in as many places as possible (to mitigate against accidental loss) and to be cryptographically authentic (to mitigate against pre-election or post-election modification). In short, it should be harder to make mistakes on election day that could prevent a provable audit after election day, and it should be easier to perform this audit after election day, even if some accidental loss has happened.

Sandler then described Auditorium, a secure, distributed logging network—his solution to this problem. In Auditorium, logs are hash chained, entangled, and broadcast. By assuming the existence of a one-way hash function, hash-chaining log entries allows an auditor to have a much higher degree of certainty about their relative order, given a certain amount of unpredictable data in each log entry. Entangling the log entries essentially makes this hash-

chaining process global across all machines in a polling place. In an entangled log, it is not the case that each machine constructs its own timeline of events: Events in one machine's timeline are necessarily dependent on events in the timelines of other machines. Because entries in the log are signed, forgery of a log event would require collusion of every machine in the network. Finally, broadcasting log events allows for replication. Because no single machine can be trusted to store everything, his design forces every machine to store everything. The result of using Auditorium on election day is a global log, stored on every machine in the polling place, whose entries can be cryptographically proven authentic and can be cryptographically reconstructed into a provable timeline.

Interesting questions from the audience indicated a bit of skepticism regarding voting machines being on a closed network. To address real-time ballot stuffing by a machine, Sandler explained that a cast ballot would not be counted unless a corresponding authorization to cast from a supervisor machine is present. This means that the supervisor machine and a voting machine would have to be in collusion. Furthermore, an act of stuffing would be noticed and logged by good machines. This is simply not possible without a network. Sandler gave a similar response when he was questioned about possible network partitions causing audit data loss: Because each machine requires an authorization to cast the ballot, this partition would be quickly noticed and rectified, causing at most the loss of one vote cycle being broadcast. Correct design would mitigate against even this, causing the broadcast of this pending vote as soon as the machine is able to reconnect to the network. Peter Neumann pressed for a better explanation of denial of service mitigation. Sandler explained that such an attack would have to disrupt every machine in order to prevent audit data from at least being replicated once.

■ ***Extending Prerendered-Interface Voting Software to Support Accessibility and Other Ballot Features***

Ka-Ping Yee, University of California, Berkeley

Ka-Ping Yee gave a summary of the extensions he made to Pvote (his prerendered user interface voting engine) to assist the visually impaired. These extensions allow the voter to *hear* as well as *see* the ballot being presented on screen. He also rehashed his justification for a prerendered user interface and small runtime code base.

Yee began by explaining that although efforts have been made to force voting machine vendors to release their source code for review, pressing for this solves only half the problem. Most voting machines in use today rely on a complex and difficult-to-audit code base. Pressing for disclosure of this code simply does not solve the problem if an audit is incredibly time-consuming and cumbersome. Yee suggested that the trusted code base needs to be explicitly small, so that an audit is simple given full source code disclosure. Furthermore, Yee emphasized that the best way to minimize the amount of trusted code is to offload determination of user interface behavior to the tool that generates the ballot. This effectively makes the user interface *prerendered*. For a given election, then, all that should be trusted is the ballot definition, with its associated user interface, and the small amount of code running on a voting machine that allows the voter to interact with and cast said ballot. As a demonstration of the simplicity of this voting machine logic Yee said that his Pvote system does just this, and it is written in 460 lines of Python.

Yee then demonstrated his system to the audience. His software did, indeed, emanate a audible recording of the screen's contents. It also emanated information regarding which option was currently selected on screen.

Comments that Yee made during his talk regarding trusting general-purpose software (such as the Python interpreter) were the cause of most of the debate during the question period of this talk. Warren D. Smith expressed specific concern with trusting software insofar as it is general-purpose: The generality of the software, Smith claims, actually proves nothing regarding its trustworthiness. Yee acknowledged that it is an open question whether or not to trust general-purpose software, but his opinion is that because the software existed before this particular application of it, and because the software isn't made specifically to run voting machines, it should be more trusted than specific voting software. Ron Rivest suggested that Java might be a better option than Python, because Java runtime environments are quite a bit more studied.

■ ***Verification-Centric Realization of Electronic Vote Counting***

Joseph R. Kiniry, Dermot Cochran, and Patrick E. Tierney, University College Dublin

Dermot Cochran gave a summary of his work, which involved his explanation of how formal specification can be used to ensure correctness of critical path voting machine modules (such as the vote counter). His talk focused on how verification-centric software engineering practice produces software that is more trustworthy. This paper's contribution is a protocol for good verification-centric software engineering practice, as well as a case study.

Cochran went over the process his team used for extending KOA, a research platform for electronic voting technologies, to support the Irish proportional representation Single Transferable Vote system, whose algorithms are legally specified. In the analysis and design phases, EBON (Extended Business Object Notation) is used to model the problem, as well as to strictly model contractual relationships between different modules of the design. These contractual bindings include preconditions, postconditions, and more general behavioral assertions. During the specification phase, this generic EBON specification is formalized using JML (the Java modeling language). JML specifications for each method and class can be formalized in JavaDoc comments, and verification that the Java code

meets the specification written in JML can be done by the JML tool chain. Currently, in the implementation and testing phase, not only can passing unit tests ensure confidence in the correctness of the code, but the code can also be tested against the formal JML specification defined in the previous phase.

Cochran showed several code snippets from his case study, highlighting Java code next to its formal JML specification.

During the question period, David Wagner pressed that some of Cochran's examples seem to hint that frequently the JML expresses exactly the same semantic concept as the Java code itself. Wagner wondered whether the JML was simply expressing the same thing and was therefore superfluous. Although admitting that this is sometimes the case, Cochran said that often it is not the case (since sometimes complex preconditions and postconditions can be more succinctly specified in JML). What's important is that the contractual specification happens before the implementation happens and that this specification can be formalized in such a way that the actual code can be checked against it.

AUDITING AND TRANSPARENCY

■ *Contractual Barriers to Transparency in Electronic Voting*

Joseph Lorenzo Hall, University of California, Berkeley
Summarized by Kyle Derr (derrley@rice.edu)

Joseph Lorenzo Hall explained his work on analyzing the contractual agreements state and local election jurisdictions make with electronic voting machine vendors, and he suggested how many of these agreements can blatantly challenge election transparency. He also made several recommendations for how these contractual agreements should be changed in the future.

First, Hall explained that this sort of research is challenging, because analysis must be done on a convenience set: Sometimes the contracts themselves are considered proprietary information, making them hard to acquire for study. Five major vendors were present in his data set of 55 contracts, whose signing dates were distributed between the years 2000 and 2006, and 82% of the contracts were with the biggest three US vendors: Sequoia, Diebold, and ES&S.

Hall organized the subset of the findings he chose to present into the following categories, where each category represents a type of transparency-barring clause. Hall found clauses that protect trade secrecy, prohibit certain types of use, discourage public record, separate escrow agreements, and limit disclosure of benchmarking and require mandatory software upgrades. Examples of trade secrecy clauses include disallowing both source disclosure and reverse engineering. In some cases *analysis* of the system is categorically prohibited. In one case, even the unit's pricing information is considered proprietary. Use prohibitions include restrictions on what hardware can be used to run the software and where (geographically) the software can be run. As for public record protection, sometimes Hall found the contract itself to be considered confidential. He also found clauses that could limit the liability of the company in the case where damages were sought because of confidential information being released as a result of legislative or judicial requirements of certain pieces of information being in public record. One of the most egregious was a mandatory upgrade provision, which forced the client to install all software upgrades no more than 10 days after their release. This, of course, means that the software upgrade would not be certified in any fashion.

Hall made several general recommendations for future clients looking to protect and encourage transparency in elections. First, contracts should always be disclosed. There is no known reason why revealing these contracts could hurt the vendor. A more pressing reason is each voter's right to know what is in them. He also suggested that limited access to source code, ballot definitions, audit logs, and vote data should be allowed, testing should not be forbidden, and damages should certainly not be limited in the case where they are incurred because of public records disclosure.

During the question session, Peter Neumann asked why only *limited* access to source code should be allowed. Hall responded that until the quality of the software improves, disclosing it to anyone who wants it will encourage attacks. Limited disclosure to trusted, third-party sources, however, cannot be stopped, as this is a useful avenue for performing audits. When asked whether any contracts speak about security obligations of the localities, Hall responded that they didn't, and that the closest thing to this is barring security analysis. When asked how long these contracts typically last, Hall responded that most clauses are in effect for the entirety of a machine's use, and in some cases, clauses do not expire (for instance, those related to proprietary information).

■ *On Estimating the Size and Confidence of a Statistical Audit*

Javed A. Aslam, Northeastern University; Raluca A. Popa and Ronald L. Rivest, Massachusetts Institute of Technology
Summarized by Kyle Derr (derrley@rice.edu)

Raluca Popa gave a summary of her simple formula for determining the number of precincts running DRE voting machines that need to be manually audited, given a desired level of confidence that no precincts have been compromised. Because it is likely that this formula will need to be computed on a hand calculator (as trusting another piece of software in elections is definitely not wanted), this formula needs to be simple and operate on few parameters. The form of audit she suggests is a simple comparison of VVPAT printouts to electronic records. The formula depends on n, the number of total precincts, b, an upper bound on how many of them could be corrupted,

and *c*, the required confidence level. Popa showed that her formula was almost exactly accurate, with error only in the positive direction, and by no more than ceil((*n* – (*b* – 1)/2).

Questions from the audience included whether the strategy assumed that there is a maximum number of two candidates per race and whether differently sized precincts were considered. Popa clarified that her formula works with any number of candidates per race and precincts of varying size: Both of these factors go into the calculation of *b*. When asked why not simply distribute a table rather than the formula, Popa claimed the table could be corrupted but that the formula is simple enough to be distributed and computed in the field. When asked if this scheme could be applied to vote-by-mail, Popa claimed that some sort of granularity would need to be applied to group the sets of cast ballots, even if it was somewhat synthetic (as would be the case in vote by mail).

■ ***Machine-Assisted Election Auditing***

Joseph A. Calandrino, J. Alex Halderman, and Edward W. Felten, Princeton University

Summarized by Elliot Proebstel (proebstel@ucdavis.edu)

Joseph Calandrino presented a short talk on Princeton's recent work in machine-assisted election auditing. Motivated by observations on electronic voting system flaws and the costs associated with traditional paper audits, the authors worked to develop software-independent auditing mechanisms that would work within a fixed budget. The authors, building on past work by C.A. Neff in 2003 and K.C. Johnson in 2004, suggest a methodology whereby ballots are machine-tallied, printed, and stored during an election. After the election, the ballots are scanned and sequentially numbered by a specialized recount machine; poll workers manually check that the recount machine correctly numbered the ballots and verify the ballot contents of a number of sampled ballots.

Calandrino also referenced the statistical research done by the team, which effectively reduces the number of ballots that must be included in a recount in order to achieve 99% confidence in the results. By implementing ballot-based audits using the authors' recommendations, Calandrino reported, the total number of ballots that would need to be audited in the Webb vs. Allen race (of Virginia's November 2006 elections) would be reduced to 2,337. This is in contrast to the 1.14 million (out of a total 2.3 million ballots cast) that would need to be audited using precinct-based auditing to achieve the same confidence. Extensions to this work include auditing only the ballots of winning candidates. In the future, the authors plan to consider cost estimates and also practical concerns, such as how to deal with errors.

Various audience members pointed out that the proposed scheme is illegal in Virginia and also that in the U.K. and New Zealand, ballot serial numbers are required by law. Someone asked about a hybrid approach that includes both precinct-based and ballot-based auditing. Calandrino responded that it depends on your definition of "hybrid." When only sampling within a precinct, the savings are smaller than expected.

■ ***An Examination of the Auditability of Voter Verified Paper Audit Trail (VVPAT) Ballots***

Stephen N. Goggin and Michael D. Byrne, Rice University

Summarized by Elliot Proebstel (proebstel@ucdavis.edu)

Stephen Goggin gave a short talk on the findings of a recent study on the auditability of Voter Verified Paper Audit Trail (VVPAT) ballots. Noting that HAVA suggests VVPAT usage, and also that 37 states legally require VVPATs, the authors looked into the difficulty of auditing the records generated by the thermal-receipt-style printers that have been installed (often retrofitted) onto Direct Recording Electronic (DRE) voting systems. The use of VVPATs has two goals: to force the voter to verify a paper copy of the ballot, and to produce a physical record for auditing. The authors assumed that the first goal was met (while mentioning that this assumption is generous and may not hold), and then set to study how error-prone and costly the auditing of such records would be.

The authors generated fake ballots, closely following the VVSG standards, trying to ensure that their ballots looked realistic. There were 120 ballots per spool, and each ballot was 2 feet long. In compliance with the VVSG standards, all ballots had "rejected" or "accepted" notation. The spools were set onto a recount fixture, and the recruited testers were given scissors and a tally sheet. These auditors (undergraduate students with good vision and fluent in English) were asked to cut apart the ballots and tally the results of a single race. After they completed this race, they were asked to go through the ballots again to tally the results for a second race.

On average, the auditors completed the first tally in 25 minutes and the second tally in 12 minutes. When generalized for full-scale elections, this represents a time requirement that is untenable for large jurisdictions. Furthermore, the error rates were particularly notable. The authors found variation from a 17% undercount to a 19% overcount. In particular, when the race being counted was lopsided, auditors tended to overcount rejected ballots at a much higher rate, suggesting that the performance of individuals counting ballots is biased by ballot contents and expectations. Auditors reported a low level of confidence in their performance, and the authors found that confidence levels did not correlate with the actual accuracy of individual auditors.

An audience member interjected to ask why 19-year-olds were recruited for this task and how their performance can be correlated to the performance of real election auditors, whose demographics were not represented in this study. Goggin replied that the authors had chosen the students as a best-case scenario: Younger participants with good eyesight were likely to be more efficient and accurate than older auditors. He concluded the talk by reporting that

these results indicate that recounts will require considerable labor and high cost, be subject to human error, and likely be influenced by auditor bias. Future work will include a comparison of auditing VVPAT records with auditing paper ballots, as well as an examination of the VVPAT usability for voters.

Q: You may need to check your assumptions about the process. Specifically, are you sure that real election auditors will separate the ballots from the spool? Also, you should look into the design of the accept and reject messages at the tail of ballots; I'm not sure your font and size choices were accurate representations. Finally, you should research the counting procedures that are actually used in jurisdictions and try to mimic those, especially with regard to your choice of having a single counter auditing the ballots. I'm not sure that's representative.

A: First, the separating of ballots from the spool made things easy for the participants. If they aren't separated in real elections, this would actually generate worse results. With regard to the font and size choices, we tried to mimic a real VVPAT by using the VVSG "large font" specifications. And we know that there are some counties that use a single counter to audit ballots.

Q: I watched the eSlate counting in San Mateo. There were three people counting. Ballots were sorted and then counted, and this was done twice. Maybe you could do follow-up work to test this?

A: Yes, that would be a good idea.

Q: Could this be addressed with machine-assisted auditing?

A: There are issues of trust. It would need to involve barcodes or op-scan systems, both of which have trust issues.

Q: Did you display undervotes or just print who was voted for?

A: We printed: [X] No Vote.

ANALYSIS II

Summarized by Elliot Proebstel (proebstel@ucdavis.edu)

- ***On the Difficulty of Validating Voting Machine Software with Software***

  Ryan Gardner, Sujata Garera, and Aviel D. Rubin, Johns Hopkins University

Ryan Gardener presented the work being done at Johns Hopkins University to investigate options available to allow poll workers to verify the authenticity of software running on electronic voting machines. The adversarial model used for this work assumes the attacker has full control of the software but is unable to make any hardware or firmware changes—even those that are seemingly benign. Gardner explained that neither direct hashing nor such hardware-based solutions as hashes signed by a TPM would suffice, because these options both ultimately rely on trusting sources that cannot necessarily be trusted to report honestly. Thus, the authors searched for a primitive that could be trusted.

Current state-of-the-art software attestation, Gardner explained, is a product called Pioneer, which is from Carnegie Mellon University. Pioneer resides in the memory and allows the verifier to provide a challenge to the system and then verify both the checksum and the time required to produce it. Because Pioneer is designed for optimal implementation, the additional instructions required to subvert the checksum process show up as overhead during runtime. Gardner's team increased the number of iterations run by Pioneer in order to magnify the attack overhead so that it could be human-measurable.

However, the authors found that even to raise the attack overhead to 3 seconds, Pioneer had to be run for 31 minutes. Gardner reported that requiring poll workers to wait for Pioneer to run for over half an hour and then detect a 3-second delay is not an acceptable solution. Furthermore, the team suspected that attacks on Pioneer-type solutions would only become more effective over time, as increased parallelization and faster CPUs incorporated into voting systems more effectively conceal attack overhead.

There were several questions about how Pioneer functioned, including about checksumming all of memory (which Pioneer doesn't do). Someone else asked whether there is a possible way to exploit the time it takes for the human to verify the long checksum? Could the checksum change itself after completion while the human is attempting to verify it? Gardner suggested reading the paper. Another person asked about the reproducibility of the timings and whether CPU temperature affect this. Gardner answered that they tested the timings at different times of day, but they did not go too far into this. Finally, someone asked whether the poll worker is supposed to use a stopwatch. Gardner said that they recommend that the poll worker use an alarm, which will go off at the expected time.

- ***An Authentication and Ballot Layout Attack Against an Optical Scan Voting Terminal***

  Aggelos Kiayias, Laurent Michel, Alexander Russell, Narasimha Sashidar, Andrew See, and Alexander A. Shvartsman, University of Connecticut

Andrew See reported on the vulnerability analysis conducted at the University of Connecticut on the Diebold AV-OS and, more recently, the AV-TSx. Previous vulnerabilities discovered in the AV-OS required access to a reader/writer for the memory card, but the authors of this work found that they could execute attacks on the AV-OS using only direct access to the device and a serial connection to a laptop.

After booting the AV-OS into debugging mode, the team was able to imitate the GEMS server with a laptop, because the AV-OS does not perform authentication on the serial connection. Using this access, the team was able to recover a dump of all memory card contents. From these contents, the team extracted the supervisor PIN and used it to enter

supervisor mode on the AV-OS. The team was then able to disable the AV-OS printer, edit communication parameters, and erase or replace memory card contents. Leveraging public knowledge about the ballot configuration, the team could use this access to remap candidate names to arbitrary locations on the ballot, allowing them to, for example, swap votes between two candidates or invalidate all votes for a given candidate. The attack code could also use cues available to it, such as the time of day and total number of ballots cast on the unit, to make an educated guess about whether it was being tested; if it suspected it was, it could report accurate results rather than attack results.

On the AV-TSx, the team developed similar attacks on the ballot layout. The text to be displayed for a candidate's name is stored in an RTF file. By swapping two RTF files on the memory card, the team could swap votes between two candidates. The database results would not match VVPAT records, but this might not be detected.

The team recommends that voting machines must ensure that ballots, VVPATs, and electronic results are consistent. This should be designed into the system, and ballot layouts should be auditable.

One audience member mentioned that Sequoia systems had been found to have similar flaws, so Sequoia has implemented the printing of (x,y) coordinates on VVPAT records. Another audience member suggested that the VVPAT could print the image from the screen in order to raise compatibility with prerendered ballots.

■ ***GEMS Tabulation Database Design Issues in Relation to Voting Systems Certification Standards***

Thomas P. Ryan and Candice Hoke, Cleveland State University

Candice Hoke presented work that was proposed in September 2006, before the team had privileged access to any GEMS (Diebold election management software) databases. The authors have subsequently had access via their work at the Center for Election Integrity in Cleveland and as public monitors for Cuyahoga County, but Hoke stressed that this work was not born of that privileged access. Furthermore, despite having participated in the California 2007 Top to Bottom Review, her contract required that she not speak of any results from that study, because the document review reports have not yet been released.

Hoke presented findings to prove that GEMS is seriously flawed in architecture and technology, resulting in data errors and erroneous results. The team outlined industry standard design requirements for database software, known as 1NF and 2NF (First Normal Form and Second Normal Form, respectively) and explained how these requirements help ensure the integrity and accuracy of database contents. Then, the team provided copious examples to demonstrate that GEMS violates both 1NF and 2NF, resulting in data errors, anomalies, and no notification that errors are occurring. Anecdotal evidence from Cuyahoga county supports these theoretical claims: Election officials reported that issuing the same query in different ways resulted in different responses, and the public audit found evidence of database corruption. GEMS uses Microsoft JET, which even Microsoft has publicly warned users is inappropriate for systems where absolute data integrity is essential.

Next, Hoke demonstrated that the federal regulatory system encourages lower standards for database design. By requiring more documentation from vendors who identify higher quality and higher horizons for database design, the federal regulatory system effectively streamlines the process for vendors with lower design standards. This reverses the incentive structure, favors low-quality design, and fails to level the field with uniform standards.

Hoke concluded with a strong call for technical experts and regulatory lawyers to forge strong partnerships in order to favor regulatory structures that will generate higher quality electoral performance and other key public functions. This partnership would focus on multiple fronts—academic, legislative, administrative, and judicial—as well as involving the media in an attempt to overcome the "fog" (glazing over of technical issues). She suggested that computer security experts are uniquely qualified for this interdisciplinary work.

The first questioner asked about avoiding the problem of regulations that suggest particular technical restrictions rather than general functionality. Hoke answered that the regulatory system structure should be discussed by both legal and technical communities. It must be played out. Hoke is in favor of a "federal floor but not a ceiling"; we should have baselines, but not limits. A second questioner commented that this work focuses on properties of relational databases instead of object-oriented databases. How can we extend this? Hoke answered that we should have discussions to find the best possible solution, weighing technical and realistic aspects. However, we cannot have federal standards that operate as a ceiling.

DESIGN II

Summarized by Elliot Proebstel (proebstel@ucdavis.edu)

■ ***Ballot Casting Assurance via Voter-Initiated Poll Station Auditing***

Josh Benaloh, Microsoft Research

Josh Benaloh presented Microsoft Research work on allowing voters to verify that their votes are being cast as they intend. Benaloh observed that attendees at VoComp (University Voting Systems Competition) in July 2007 were confused about the verifiability of all systems except the DRE. It wasn't verifiable, Benaloh noted, but they trusted it because they understood or recognized it. This raised the challenge: Can an open-audit voting system be built with full end-to-end verifiability without trusting the software and look like a DRE? Benaloh claimed that it can be done, but the details are difficult to implement.

Benaloh reported that we have solid protocols for taking a set of encrypted ballots and verifiably processing them to produce an accurate tally. The transformation of encrypted ballots to a tally is a black-box process that requires no trust in software, hardware, or people. Thus, we can solve the "counted as cast" problem. The "cast as intended" problem is not so easy. Voters need to be able to ensure that machines are allowing them to cast ballots as intended. Clever ideas recently developed forcibly engage voters in the verification process, but these are mostly still too cumbersome for users. What happens if voters are allowed but not required to check ballot validity? The principal requirement is that a voting device cannot know the identity of the user. The user of the voting device need not even be a qualified voter; it could be an election official or a suspicious voter.

Benaloh explained a first effort where voting devices are isolated, with stand-alone units charged with capturing a voter's intentions and turning those intentions into encrypted ballots. The voter can either cast this encrypted ballot or have it decrypted to verify that it has been properly formed. As long as the selection of which ballots will be challenged is unpredictable by voting devices, it takes very few challenges to obtain extremely high confidence. With lots of voters, even a small percentage of voters who challenge their ballots results in very high confidence. Some problems remain with this scheme, but Benaloh reported that it looks promising. The scheme is this: A voter walks into a poll station and (if legally required) provides ID to a poll worker. The voter receives a token indicating the correct ballot type. The voter inserts the token into the voting device and makes selections. The voter receives an encrypted ballot. At this point, the voter can (1) provide the ballot to be cast or (2) have the device open the ballot via challenge. This should be unobtrusive. After the selections are made, a voter can be asked, "Do you wish to cast this vote?" If the voter chooses "yes," the device digitally signs the encrypted ballot to indicate its eligibility for casting, and the voter is instructed to take the vote to a poll worker; the poll worker scans the encrypted vote and gives the voter the original as a receipt. If the voter selects "no," the device provides a verifiable decryption which the voter may take home.

The encryption is deterministic. Encryption of ballot b is performed by selecting a random value r and forming the encryption $V = E(b,r)$. The voting device reveals a vote V by revealing b and r; a voter can take this home and verify it on his or her own computer. Most voters probably won't, but at least they can. Ballot protection is ensured by printing the encrypted ballot before the voter indicates whether or not it is to be challenged, but the specifics of the encrypted ballot cannot be known to the voter before the choice is made, in order to avoid voter coercion. Chain-voting is still possible within this scheme and needs to be addressed. Remote voting, however, has very substantial coercion problems, but many people want it anyway. Benaloh reported that the proposed scheme could even be supported in Internet voting. The bottom line, according to Benaloh, is this: Adding a single question to the end of the voter process can add verifiability.

Someone asked whether the system could still cheat by decrypting "incorrectly." Benaloh replied that you cannot prove that the machine is recording incorrectly. You need to be able to check in real time, online. Then someone asked how the voter can tell that the number r is really random. Benaloh answered that the voter can't know this, and that I can't prove privacy, but neither can you. In response to whether there can be any protection against coercion, Benaloh answered, "No, you can't prove absolute privacy, just take good steps."

■ ***Bare-Handed Electronic Voting with Pre-processing***

Ben Riva and Amnon Ta-Shma, Tel-Aviv University

Amnon Ta-Shma presented an end-to-end scheme for election verification that is intended to allow voters to vote "bare-handed," that is, without bringing their computers to the voting booth. This should ideally be as simple as a DRE but with cryptographic guarantees. Ta-Shma reviewed previous work from Chaum and Neff and indicated that neither of those schemes provides the voter with privacy against the booth or the encryptor. This is a primary goal that the team from Tel-Aviv University is seeking to meet; the voter should be able to prepare his or her own ballot, without having to trust anybody else, while still being able to vote bare-handed. This leaves a quandary: If voters prepare their votes at the voting booth, they must bring a computer, but if they prepare their votes at home, they can be subject to coercion.

The authors' work seeks to avoid these problems and pitfalls. There are three primary advantages of pre-processing: (1) The voter can use open-source public code; (2) The voter can use any computer hardware; (3) The scheme is coercion-resistant, so a voter can get his or her ballot encrypted by a friend, a government machine, a coercer, or a political party. The voter still gets privacy as long as the party preparing the ballot does not maliciously cooperate with the voting booth. This allows voter to choose what level of privacy is desired.

The protocol works as follows: A voter comes into the booth with a ballot for each candidate and chooses which one to use at the booth. The booth uses a cut-and-choose test to ensure that each voter comes with a ballot for each candidate and that the voter can match ballots to candidates. Each ballot has two sides, a front and a back. On the front side, the ballot is in plain text; on the back, it is encrypted. Both front sides are published. A poll worker randomly chooses one ballot, and its back side is also published. The booth re-encrypts the front side of the remaining ballot twice and prints it, covered with a scratch surface. The voter chooses a candidate from one column and uses the other column for testing the booth. To test the

voter, auditors check that the published back side matches the published front side. Every candidate appears exactly once, and the encryptions on the front side match the candidates on the back side.

The booth needs to re-encrypt in order to prevent a coercer from having full information. If the booth prints the re-encryptions without a scratch surface, vote-buying is possible. For example, a coercer could say, "Vote using a re-encryption that starts with 110 and get $100," forcing a random vote. We want the vote to be independent of the encrypted strings. This scheme: (1) provides unconditional unforgeability even against all-powerful adversaries (common also to other crypto schemes); (2) is receipt-free: outsiders only see the encrypted vote and the revealed column; (3) provides coercion-resistance, because there is a probability of 1/2 to coerce a voter without being caught, which deters large-scale coercion (as there should be a significant risk attached to coercion); and (4) allows the voter to vote bare-handed. Moreover, it is a modular scheme, and it can be based on several existing schemes. This protocol transfers the ballot preparation from the booth to the voter (at a pre-processing stage) and the tallying is unchanged. Future directions for this work include simplifying the scheme and relaxing the assumptions—mainly the assumption that the public board is readable from anywhere.

The first questioner observed that at VoComp, most people thought cryptography was a study of where dead people go (crypts). The population won't understand this if it involves cryptography. Ta-Shma responded that they ask the voter to come in with two ballots. The voter doesn't have to understand crypto; only the auditors do. Someone pointed out that there are already variants that deal with booth-trust issues. Ta-Shma agreed, but said that their contribution is the use of multiple ballots. Someone asked what happens if the poll worker colludes with a coercer. Ta-Shma conceded that the scheme fails in that case. Finally, someone expressed this concern about the crypto process: The public is the auditor. Ta-Sham answered that the crypto behind the scheme is very simple and easy to understand. All you have to do is calculate some function, and the software to do it can be downloaded from the Internet.

■ ***Three Voting Protocols: ThreeBallot, VAV, and Twin***

Ronald L. Rivest, Massachusetts Institute of Technology; Warren D. Smith, Center for Range Voting

Ron Rivest presented some "outside of the box" ideas on end-to-end voting systems, which he claimed were a most promising general direction for election verification. The most common end-to-end systems are based on cryptography, but he was able to present three options for doing it without crypto: ThreeBallot, VAV, and Twin.

In ThreeBallot, each voter casts three plain text ballots. All three go on a public bulletin board (PBB). The voter takes home a copy of an arbitrarily chosen one as a receipt. It doesn't indicate how he or she voted, but serves as an integrity check on the PBB. Every ballot has a serial number that is not easy to remember but is easy to type. Each row of a ballot has at least 1 mark, not 0 and not all 3. Each candidate gets n extra votes (where n = number of voters) but the election outcome is the same. This works for everything except rank-order choices or write-ins. Votes are cast in a physical ballot box. The order of ballots is random and is not tied together, but a machine checks before casting to ensure that ballots are valid: It only checks, it doesn't tally. The voter arbitrarily gets to choose one as a receipt, and no record is kept of which was the receipt. Receipts should be unforgeable. The voter confirms the posted ballot on the PBB after the polls close. Each ballot has a unique ID, so it can be located. Voters should not see (and/or be able to memorize) IDs for ballots that were not copied (to prevent vote-selling.) Plain-text ballots are subject to short ballot requirements, to prevent reconstruction attacks. Since an attacker doesn't know which ballots posted on the PBB have copied receipts, any significant tampering is likely to be detectable. The use of three ballots makes this coercion-free. Voters can't sell their votes by using their receipts. Using only the PBB and voter receipts, neither an adversary nor a voter can determine which three ballots were in an original triple. However, the usability is not so good, and the system is confusing to many. It would be possible to mix "OneBallot" (ordinary ballots) with ThreeBallot, but no receipts could be issued. End-to-end security provides voter confidence; the voter can check that his or her ballot is included in the tally and can check that collection and tallying are done correctly, all without crypto.

Rivest next presented VAV (Vote/Anti-Vote/Vote), in which the voter casts three ballots and takes a copy of one home as a receipt, but one ballot must cancel another. The anti-vote ballot is marked as "ANTI," so the voter casts one ballot the way he or she wants, another ballot the way he or she doesn't want, and an anti-vote to cancel the unwanted one. The tallier finds and removes pairs of ballots that cancel one another and only counts the remainders. This handles any voting system.

In Rivest's final scheme, known as Twin, the voter gets to take home a copy of somebody else's ballot. The voter can verify it from the PBB. All original ballots are put into a bin as they are cast, and every voter (after the first ten) is given a copy of an arbitrary ballot from the bin. Voters cannot prove their own ballot and don't know whose ballot they have. An attacker cannot collect all copies of any chosen receipt, because receipts are given with random selection, using replacement. A constant fraction of all receipts are taken home with high probability. Rivest concluded that it is possible to implement end-to-end security without crypto, and end-to-end schemes provide improved assurance of correctness of an election outcome.

The first question was about how to get an unforgeable receipt without crypto. Rivest answered, "Maybe watermarked paper? A digital signature?" The next person pointed out that, with VAV, people are more likely to take home the one that's the actual ballot. Rivest explained that

this is the same in other schemes, but the voter has deniability here, which helps prevent coercion. The next questioner had a stumper: What prevents voters from casting two ballots for the candidates of their choice and then an Anti-Vote ballot against the candidates they oppose? Rivest confirmed that there isn't a solution to that problem yet. Someone else wondered, if ballots are posted in plain text, what prevents stray marks from allowing vote-selling? Rivest said that the posted records should be digital versions of the plain text. Finally, someone pointed out that posting ballots on a bulletin board is new. What new opportunities does this present for wholesale fraud? Rivest answered that new kinds of verifiability will help in detecting and preventing wholesale fraud. This is a whole new layer of defense.