Election Verification Technology

Threat Analysis

Revisions

Revision	Description	Date	Author
1.0	Baseline	9/15/03	R. Green / J. Adler

Table of Contents

Introduction

Americans have trusted their election systems for more than two hundred years. Over the past few years, however, doubt has crept into the mainstream American consciousness fueled by the 2000 Presidential Election and several other election irregularities.[1]

Such irregularities demonstrate that all election systems are vulnerable to unintentional errors, manipulation and malicious attacks. This is true of both electronic and paper-based voting systems. The introduction of computerized voting and tabulation machines has created the possibility of large-scale fraud by attacking a single location such as the vendor's software development facility.

However, paper-based systems are also vulnerable because the ballot readers used in modern paper-based systems are programmable. Although hand counting can be used in a close race, it is not practical for a vast majority of elections. Even when allowed, hand counting does not eliminate the dangers of ballots being added, modified or destroyed. As illustrated during the 2000 election in Florida, such fraud in a single jurisdiction can affect the outcome of a close Presidential election.

Voting systems underpin the peaceful transition of power, so it is essential that the electorate have trust in the systems that elect its leaders. Trust in any election system is built upon the system's ability to count the intentions of voters when faced with a myriad of threats, both internal and external. An open and transparent threat assessment is crucial to the electorate's trust in their election system.

A good threat assessment evaluates (1) assets, (2) potential attackers to those assets, and (3) vulnerabilities of those assets to potential attackers. Defensive countermeasures must thwart attacks to those vulnerabilities.

For each vulnerability, this threat analysis details appropriate defensive countermeasures. These countermeasures may prevent, detect, or deter attacks depending on the asset being defended.

Since there is no comprehensive formula for threat analysis, it is generally accepted by security experts that publicly vetted, open systems are most secure since they allow for broad, collaborative analysis. It can then be understood on what assumptions and properties the system's security rests. Since public elections administer the peaceful transfer of power through transparent consent of the governed, transparency is even more vital.

---

[1] *Computer Voting Is Open to Easy Fraud, Experts Say,* New York Times, July 24 2003;
Broward officials misplace 103,222 votes, but outcomes are unchanged, Sun Sentinel, November 7 2002;
Cartridge problem casts pall on Palm Beach County's new voting machines, Sun-Sentinel, March 14 2002;
Machines lose 294 early votes -- Software glitch means Wake voters will get re-vote, North Carolina News Observer, October 31 2002;
Firm admits errors in counting votes for Hawaii, Venezuela, Honolulu Star-Bulletin, June 7 2000.

Assets and Attackers

For elections, the principal asset is governmental power. That power is transferred by the results of counting voted secret ballots. Hence, integrity of the voted ballot is critical through the entire process from capturing the voter's intent, casting it into the ballot box, counting it to produce the election results, and finally retaining it to resolve disputes.[2]

Given that the primary asset of an election is governmental power, it must be assumed that attackers are highly motivated, well financed, sophisticated, and could be outsiders as well as insiders with full knowledge of the election system. These attackers could be political operatives, voters, vendor personnel, polling place workers, election administrators, or just pranksters.

It is assumed that the most virulent attacker is extremely motivated, sophisticated and well financed[3]—even a rival government could mount the attack. Furthermore, it is assumed that insiders are not trusted since the attacker could compromise them as well. In fact, in the case of a dictatorship, attacks could be mounted from the incumbent regime.[4]

Vulnerabilities

The *principal vulnerabilities* to the voted secret ballot are (1) undetected compromise of election integrity, (2) compromise of ballot secrecy, and (3) denial of voting service.

An undetected compromise of election integrity is the most horrific vulnerability because it results in an unwarranted change of political power. In contrast, a detectable compromise reduces to a denial-of-service attack against the system. For example, the September 11, 2001 terrorist attacks occurred on Election Day in New York. Because many voters were denied the mechanism of voting, the election was disqualified and postponed. However, the principal asset (i.e., governmental power) was not erroneously transferred and, thereby, not compromised.

This analysis does not attempt to classify or defend against all types of recoverable "denial of service" attacks since all election systems suffer from such attacks. Typically, such attacks are defended by procedural countermeasures, such as L&A testing, which ensure that election machinery is maintained and well configured for Election Day.

---

[2] Obviously, there are other less vital assets, such as costs of rerunning an election, equipment theft, etc. However, this analysis is most concerned with the principal asset of governmental power and the secret voted ballots that enable its peaceful transfer.

[3] According to the Federal Election Commission, over $709 million in contributed and matching funds was raised for the 2000 Presidential election. See http://herndon1.sdrdc.com/fecimg/srssea.html

[4] *International Community denounces Zimbabwe elections*, CBC News, 13 March 2002, http://www.cbc.ca/stories/2002/03/13/zimvote020313.

Most vulnerabilities fall under one of the above three principal vulnerabilities. For example, undetected election compromise could result from either incorrect capturing of voter's intent (whether inadvertent or malicious), modification of the cast votes in the ballot box, and/or changing the counted election results.

All other vulnerabilities are subordinate to these higher-level, critical vulnerabilities. Below are a few example vulnerabilities that can result in undetected compromise of election integrity:

- Unauthorized modification of the election system;
- Alteration of voting system audit trails;
- Alteration of the capturing of a vote;
- Introduction of duplicate or fraudulent votes;
- Alteration of system data stores, including ballot definitions, collected votes and calculated vote totals; or
- Unauthorized access to system data stores, including individual votes, vote totals, audit logs, and system authentication information.

Similarly, the following vulnerabilities can result in compromise of ballot secrecy:

- Unauthorized modification of the system;
- Any access, authorized or otherwise, to information that may link voter identification data to the content of specific votes cast by a voter.

Historically, elections have been conducted under public observation so that the principal vulnerabilities cannot be exploited and the consent of the governed can be transparently transferred. As populations have grown, modern election procedures have delegated this observation function to a group of election observers, or trustees, with politically diverse allegiances to ensure that the election is conducted with utmost integrity.

Defensive Countermeasures

Defensive countermeasures are designed to thwart attacks that exploit the principal vulnerabilities in attempts to compromise the assets. Countermeasures can be designed into the data protocol, the software/hardware implementation, and/or the election administrative procedures.

Countermeasures typically fall into three categories: *protection*, *detection*, and *deterrence*. Protection defends the vulnerability by *stopping* an attack. A bank safe is a good example of a protection countermeasure. Detection defends the vulnerability by *raising an alert* when an exploit is attempted. A vault sensor is an example of a detection countermeasure. Deterrence defends by reducing the impetus for an attacker to exploit the vulnerability—often by *raising*

the punishment in the event of detection. An armed bank security guard is an example of a deterrence countermeasure.

While stopping an attack through protective countermeasures is the most desirable, it is impossible to implement perfectly. For example, the countermeasures protecting a bank safe can be breached by either cracking the combination, coercing an insider to reveal the combination, or blowing open the vault door.

Given the assumed tenacity, vast resources and sophistication of an election attacker, protective countermeasures are insufficient to defend election integrity. No election system is absolutely impervious to attack, especially from trusted insiders who have full knowledge of the election system's architecture and procedures. Typical countermeasures used to protect election systems include:

- Putting locks on doors;
- Restricting access to assets;
- Data encryption;
- Establishing and maintaining controls to ensure that accidents, inadvertent mistakes and errors are minimized;
- Hiring "trusted" personnel to prevent intentional manipulation, fraud or malicious mischief.

Unfortunately, all of these countermeasures can be defeated, many of them easily and without detection. When these imperfect "Fort Knox" countermeasures fail, the principal asset (i.e., transfer of governmental power) can be compromised without detection.

Similarly, while deterrence is desirable, it has proven to be only partially effective. For example, long prison sentences haven't eliminated bank robberies. The threat of spending many years in jail doesn't help if the election is compromised without detection.

In the voting context, any attack should, at a minimum, be *detected* to defend against election compromise. Protection and deterrence alone are no defense if a breach results in undetected election compromise. Detection is the most essential and effective countermeasure for election systems because it prevents unwarranted transfer of governmental power and allows recovery from the attack.

Voter Verification and Results Verification are the cornerstones of an effective detection countermeasure strategy for election systems. By giving the voter the means to determine whether his/her intent has been captured accurately and stored in the ballot box, and giving any observer the means to determine if the results have been tabulated correctly from the ballot box, any breach of election integrity can be detected. Our VHTi protocol provides electronic voting systems with the means for Voter Verification and Results Verification.

VHTi Overview

The system described principally relies on a set of election trustees, cryptographic key management, public commitment of vital data, and independently verifiable operations. See [NA03] for technical overview; [Nef03] and [Nef03_1] for mathematic detail; and [VH03] for definition of common terms and data structures.

A distributed group of election trustees, arbitrary in size and affiliation, is entrusted with ballot secrecy. This is similar to requiring multiple keys for launching nuclear missiles—a threshold number must act in concert for successful operation. Since the number and affiliation of trustees can be defined before the election, the level of security can be made arbitrarily high by selecting a sufficient number of trustees with divergent or opposing agendas to make collusion highly improbable. For example, trustees can be selected from all political parties on the ballot, the press, watchdog groups, non-profit organizations, etc.

The analysis assumes that insiders (e.g., vendors, poll-workers, election administrators) can be compromised and are in a unique position to violate election integrity and voter privacy. It also assumes that all computers used in the election, including the voting machines, poll worker machines, tabulation machines and vendor software development machines, can and will be compromised.

Cryptography relies on management of secret data (e.g., private cryptographic keys and codebook keys). This data must be securely generated and transported. Public commitment of key data ensures that any later modification of that data is easily detected and cheating, thereby, discovered.

Voters, observers, officials, and trustees must have the means and opportunity to verify correct and honest operation through immutable evidence. As an example, the voting machine itself (which, as mentioned above, is not trusted) should be audited during voting. Therefore, it is critical that the machine not be aware that it is, in fact, being audited. The assumed untrustworthy machine would only choose to cheat when not being tested. Pre-election and post-election “logic and accuracy” tests, standard procedure in today’s election operations, would not detect such malicious operation.

Threat Analysis Overview

This analysis identifies three levels of countermeasures to defend against identified vulnerabilities:

1. The **VHTi (Protocol) Level** contains defenses to make the underlying election data secure, transparent, and auditable;

2. The **Implementation Level** contains defenses to make the voting machine and surrounding hardware and software secure; and

3. The **Procedural Level** contains defensive procedures and processes for conducting a secure election.

The analysis focuses most directly on the VHTi (Protocol) Level, and only generally discusses the Implementation and Procedural Levels. It is important that both the implementation and election procedures not undermine the verification defenses provide by the protocol.

Although VoteHere makes recommendations for the Implementation Level, it is determined and controlled by DRE manufacturers who integrate the VHTi component software into their products. For example, the VHTi protocol depends on secret cryptographic keys to protect voter privacy, and the underlying key management implementation must be vetted to ensure that those secret keys are managed appropriately to avoid compromise. The intent of the Implementation Level discussion is to highlight the security requirements and trade-offs a DRE vendor needs to make. It is at this level that customer and certification requirements enter the design, forcing possible trade-offs between security and usability.

It is important to note that, except for stuffing the ballot box, which can only be prevented by publishing the voter roll, election integrity under the VHTi protocol does not depend at all on the underlying implementation in the DRE or election management computer. The data is completely independent of the computer software so that voters and election observers can detect any alterations.

Furthermore, while VoteHere also makes recommendations for the Procedural Level, as do DRE vendors, it is determined and controlled by each jurisdiction. The laws, policies and practices governing election procedures typically vary by jurisdiction and may expose the system to certain attacks. For example, in order to detect ballot box stuffing, polling place procedures must publish the checked-off voter roll—this is true for all election systems, not just those using electronic devices.

In order to facilitate the analysis, a generalized DRE reference design based on currently available DRE voting devices is assumed. To be clear, the analysis does not cover a specific software or hardware implementation or set of election procedures. An implementation and procedural evaluation of a VHTi-based DRE must be completed to ensure that any implementation and procedural decisions do not undermine the defenses provided by VHTi.

The generalized DRE reference implementation used here makes these assumptions:

1. Voted ballots are captured in multiple internal memory locations;

2. Voted ballots are captured on memory cards and transported to central jurisdiction for counting either manually, electronically, or both; and

3. There is no cryptographic protection from ballots being changed.

Attack Tree

Attack trees provide a way to capture threats against a system [SSW02][Sch02]. Each root of the tree shows the goal of the attack, while the branches provide a method to meet the goal.

For each attack, a countermeasure is provided that defends against the vulnerability. Countermeasures referenced in the tree are summarized in the *Defensive Countermeasures* section. Further detail can be found in the *VHTi Overview* section, [Nef03], [Nef03_1], and [NA03]. Terms are defined in [VH03].

The natural nesting structure of the attack tree provides a way to capture attacks, costs of mounting attacks, and countermeasures. For example, consider the following attack tree excerpt:

ATTACK	COUNTERMEASURE
1. Goal: Compromise election integrity undetectably	V1 V2 V3 I1 I2 I3 P1 P2
1.1. Capture other than voter intent (OR)	V1 V2 V3 I1 I2 I3
1.2. Change ballot box contents (OR)	V1 V2 V3 P1 P2
1.3. Change election results	V1 V2 V3 I1 I2 I3 P1

The goal of "Compromising election integrity undetectably" can be accomplished by (1) capturing other than voter intent; (2) changing ballot box contents; or (3) changing election results. Protocol, implementation, and procedural countermeasures are needed to thwart these attacks.

Expanding the tree further shows where the countermeasure is applied to a specific attack. For example, one of the ways to "change ballot box contents" is to add a ballot by altering the voter roll. As noted under *Threat Analysis Overview*, only procedural countermeasures can ensure that only legitimate voters are given access to the ballot. The VHTi protocol and voting machine implementation cannot defend against this attack.

Finally, this analysis makes the familiar assumption that the attackers are highly motivated, sophisticated and well financed. These attackers can be either outsiders or insiders with full system knowledge. The analysis assumes that outsiders can conduct many of the same attacks as insiders even though the relative cost of executing an attack is higher because they must gain access first—that is, insiders are considered outsiders with passwords and badges. The analysis highlights attacks where the distinction between outsider and insider is notable, though the countermeasure for any given attack by insider or outsider is likely the same.

The following is an abbreviated attack tree that discusses the outermost attacks and defensive countermeasures. Countermeasures that defend against the outermost attacks, by definition, defend against inner attacks. The full attack tree is described in the *Appendix B: Detailed Attack Tree*.

Abbreviated Attack Tree

ATTACK	COUNTERMEASURE
1. Goal: Compromise election integrity undetectably	V1 V2 V3 I1 I2 I3 P1 P2
1.1. Capture other than voter intent (OR)	V1 V2 V3 I1 I2 I3 P1 P2
1.1.1. Modify software or election configuration (AND)	V1 V2 V3 I1 I2 I3
1.1.1.1. Modify software on vendor's development, archive or testing computer (OR)	V1 V2 V3 I1 I2 I3
1.1.1.2. Modify object code after it is loaded into target computer (OR)	V1 V2 I1
1.1.1.3. Modify election configuration after it is loaded into DRE or election management computer	V2 V3 I1 I2 I3
1.1.2. Fool L&A testing	V2
1.1.2.1. Modified software acts honestly for pre-election and post-election L&A test (AND)	V2
1.1.2.2. Modified computer activates rogue "payload" for actual election	V2
1.2. Change ballot box contents (OR)	V1 V2 V3 I1 I2 I3 P1 P2
1.2.1. Change voted ballot (OR)	V1 V2 V3 I1 I2 I3
1.2.1.1. Capture other than voter intent (1.1) (OR)	V1 V2 V3 I1 I2 I3
1.2.1.2. Modify ballots in ballot box	V1 I1
1.2.2. Add ballot (OR)	I1 I2 I3 P1 P2
1.2.2.1. Use voting ticket to vote multiple times (OR)	I2
1.2.2.2. Use DRE to vote using illegally obtained voting ticket (OR)	I1 I2 I3 P1 P2
1.2.2.3. Generate ballot and add to ballot box	V3 I1 I2 I3 I4 P1 P2
1.2.2.4. Tabulate before the polls close and target "get out the vote"	V3.1
1.2.3. Delete ballot (OR)	V1 I1
1.2.3.1. Modify DRE software (1.1.1) (OR)	I1
1.2.3.2. Modify ballots in ballot box (1.2.1.2)	I1
1.2.3.3. Assert voter forged the verification receipt if discrepancy produced	V1
1.2.4. Delete ballot and add different ballot (same as Delete (1.2.3) plus Add (1.2.2) or Change (1.2.1))	
1.3. Change election results	V1 V2 V3 I1 I2 I3 P1 P2
1.3.1. Change encrypted ballot box contents (OR)	V1 V2 V3 I1 I2 I3 P1 P2
1.3.2. Change decrypted ballots (OR)	V3.5 I1
1.3.3. Publish false tally and false election transcript	V1 V2 V3 I1
2. Goal: Compromise voter privacy	V3 I1
2.1. Determine voter's selections from published verification statement (OR)	
2.1.1. Match voter with published verification statement (AND)	I1
2.1.2. Generate codebook for BSN in verification statement	
2.2. Modify DRE software to capture and record voter's selections (OR)	I1
2.2.1. Modify DRE software to record unencrypted voter selections (1.1.1) (AND)	I1
2.2.2. Record voter names in order that they voted (2.1.1.3.1)	

Defensive Countermeasures

VHTi (Protocol) Countermeasures

VHTi provides these Protocol Level countermeasures. See the *VHTi Overview* section and [NA03] for technical discussion. Detailed mathematic protocols can be found in [Nef03], and [Nef03_1]. Definition of common terms and data structures can be found in [VH03].

V1. Voter-verification allows voter to ensure DRE accurately captured intent [Nef03_1] (AND)
- Result: Voter can verify ballot was cast as intended
- V1.1. Codebook displayed (committed) prior to voting ballot (AND)
  - Result: Voter can detect if DRE cheats by changing codebook
- V1.2. Verification receipt printed before ballot is cast (AND)
  - Result: Voter can verify that correct codes were recorded
- V1.3. Verification receipt signed after ballot is cast (AND)
  - Result: Voter can prove DRE cast ballot
- V1.4. Election transcript published after election
  - V1.4.1. Publish verification statement for each ballot
    - Result: Voter can compare verification receipt with verification statement

V2. Observer-verification allows DRE to be randomly audited during election [Nef03_1] (AND)
- Result: Modification of DRE is detectable by election observer
- V2.1. Ballot definition digitally signed and published pre-election (AND)
  - Result: Observer can ensure blank ballot committed/loaded accurately
- V2.2. Ballot Sequence Number (BSNs) published pre-election (AND)
  - Result: Observer can ensure only valid BSNs are generated by DRE
- V2.3. Codebook commitments signed and published pre-election (AND)
  - Result: Observer can verify codebook matches published commitment
- V2.4. Election parameters and codebook outputted (i.e., committed) before ballot voted
  - Result: Observer can detect change in election parameters or codebook

V3. Results-verification allows anyone to audit ballot box and tabulation process [Nef03] (AND)
- V3.1. Use Pedersen secret sharing protocol to generate trustee private key (AND)
  - Result: Voter privacy (voted codebooks) protected
  - V3.1.1. Recruit sufficient number of trustees with divergent or opposing agendas
    - Result: Voter privacy is protected
    - V3.1.1.1. Recruit opposing party reps, press, League of Women Voters reps, etc.
      - Result: Trustees unlikely to collude
    - V3.1.1.2. Recruit sufficient number of trustees to minimize probability of collusion
      - Result: Trustees unlikely to collude
- V3.2. Compute answer marks from blank ballot data (e.g., candidate name)
  - Result: Answer marks can't be reassigned to different answers
- V3.3. Use Neff Shuffle protocol to tabulate ballots (AND)
  - Result: Ballots cannot be changed during tabulation without detection
  - Result: Mathematically provable audit trail from tally back to ballots
  - Result: Anyone can audit the tabulation process
  - Result: Privacy is protected
- V3.4. Publish encrypted voted ballots (AND)
  - Result: Anyone can verify only valid and unique ballots are in ballot box
- V3.5. Publish election transcript (AND)
  - Result: Anyone can prove tally resulted only from ballots in ballot box
- V3.6. Publish trustee codebook secrets for unvoted BSNs
  - Result: Observer can verify DRE codebook values match answer marks
- V3.7. Use published data and observer-gathered data to verify election results
  - Result: Any discrepancy between DRE config and transcript is detected
  - Result: Integrity of election is Information Theoretic Secure

Implementation Countermeasures

These Implementation Level countermeasures are provided as guidelines to the application that integrates VHTi. They illustrate areas that require care by the implementation in order to maintain the defenses provided by the VHTi protocol countermeasures.

I1. Use secure key generation, management, and distribution techniques
- Result: Forging key is infeasible
- I1.1. Use cryptographically acceptable random number generators
  - Result: Key can't be determined through RNG pattern
- I1.2. Use PKI key generation
  - Result: Factoring discrete logarithms based on large primes is infeasible
- I1.3. Use secure key management techniques
  - Result: Key protected from theft on generating computer or target computer
- I1.4. Use secure key distribution techniques
  - Result: Key protected from theft during distribution

I2. Each authorized voting ticket can vote only one ballot
- Result: Only one vote per voter
- I2.1. Generate unique random number (voting number) (AND)
  - Result: Each voter has unique number that can't be predicted
- I2.2. Sign voting number with poll worker private key (AND)
  - Result: Validates voting number and links it to poll worker
- I2.3. Write signed voting number to voted ballot (AND)
  - Result: Voting ticket enabled
- I2.4. DRE rejects subsequent use of same voting number (AND)
  - Result: Only one vote per voting ticket
- I2.5. Election management computer rejects subsequent ballot with same voting number (AND)
  - Result: Only one vote per voting ticket
- I2.6. Published election transcript contains signed voting number for each ballot
  - Result: Anyone can tell if DRE or election management computer modified

I3. Each ballot can be traced to the poll worker who authenticated the voter
- Result: Provides audit trail against corrupted poll worker
- I3.1. Write signed voting number to verification receipt (AND)
  - Result: Verification receipt can be traced to authorizing poll worker
- I3.2. Write signed voting number to voted ballot (AND)
  - Result: Ballot can be traced to authorizing poll worker
- I3.3. Published election transcript contains signed voting number for each ballot
  - Result: Verification statement can be traced to authorizing poll worker

I4. Each Ballot Sequence Number (BSN) may be voted only once
- Result: Only one vote per BSN
- I4.1. DRE rejects subsequent use of BSN (AND)
  - Result: Only one vote per BSN
- I4.2. Election management computer rejects subsequent ballot with same BSN (AND)
  - Result: Only one vote per BSN
- I4.3. Published election transcript contains BSN for each ballot
  - Result: Anyone can tell if DRE or election management computer modified

Procedural Countermeasures

These Procedural Level countermeasures are provided as guidelines to the jurisdiction that operates the DRE system incorporating VHTi. Local law often governs election procedures. Hence, these procedures are not meant to be comprehensive but illustrate areas that require care in order to maintain the defenses provided by the VHTi protocol and implementation countermeasures.

P1. Authenticate each voter at poll site (AND)
- P1.1. Require voter ID (AND)
  - Result: Voter identified before obtaining ballot
- P1.2. Require voter's live-ink signature on voter roll
  - Result: Impersonators can be caught, legitimate voters cannot repudiate

P2. Publish voter roll with signatures after election
- Result: Must be same number of signatures as ballots
- Result: Addition of invalid voters detected, impersonators caught

P3. Voter draws random voting ticket (I2) from drum or basket
- Result: Poll worker cannot associate voter with voting ticket number in ballot
- P3.1. All voting tickets for given poll worker generated in advance
  - Result: Poll worker doesn't know which voter will get ticket
- P3.2. Voter draws random voting ticket from drum, barrel or basket
  - Result: Poll worker doesn't know ticket number drawn by voter

P4. Let voters discard verification receipts in poll site trash can and let any voter take them
- Result: Buyer/coercer can't be sure voter generated verification receipt

P5. Have stacks of random printed codebooks freely available in poll site
- Result: Vote buyer/coercer can't be sure captured codebook was used

P6. Have photos of on-screen codebooks freely available on-line
- Result: Vote buyer/coercer can't be sure captured codebook was used

P7. L&A testing before and after election
- Result: System configuration in a known state.

P8. Physically secure poll site
- Result: Poll site is in known state

Appendix A: References

[NA03] C. A. Neff, J. Adler. Verifiable e-Voting. *VoteHere, Inc.*, http://www.votehere.net/vhti/documentation/verifiable_e-voting.pdf, August 2003.

[Nef01] C. A. Neff. A Verifiable Secret Shuffle and its Application to E-Voting. *Proceedings of the 8th ACM Conference on Computers and Communications Security (CCS-8)*, November 2001.

[Nef03] C. A. Neff. Verifiable Mixing (Shuffling) of ElGamal Pairs. *VoteHere, Inc.*, http://www.votehere.net/vhti/documentation/egshuf.pdf, August 2003.

[Nef03-1] C.A. Neff. Detecting Malicious Poll Site Voting Clients. *VoteHere, Inc.*, http://www.votehere.net/vhti/documentation/psclients.pdf, September 2003.

[SSW02] C. Salter, O. Saydjari, B. Schneier, J. Wallner. Toward A Secure System Engineering Methodology. *New Security Paradigms Workshop*, http://www.counterpane.com/secure-methodology.html, September 1998.

[Sch02] B. Schneier. Attack Trees: Modeling Security Threats. *Dr. Dobbs Journal*, http://www.counterpane.com/attacktrees-ddj-ft.html, December 1999.

[VH03] VHTi Terminology. *VoteHere, Inc.*, http://www.votehere.net/vhti/documentation/terminology.pdf, September 2003.

Appendix B: Detailed Attack Tree

ATTACK	COUNTERMEASURE
1. Goal: Compromise election integrity undetectably	V1 V2 V3 I1 I2 I3 P1 P2
1.1. Capture other than voter intent (OR)	V1 V2 V3 I1 I2 I3 P1 P2
1.1.1. Modify software or election configuration (AND)	V1 V2 V3 I1 I2 I3
1.1.1.1. Modify software on vendor's development, archive or testing computer (OR)	V1 V2 V3 I1 I2 I3
1.1.1.1.1. Gain access to computer (AND)	I1
1.1.1.1.1.1. Obtain privileged access information for user or administrator account (AND)	I1
1.1.1.1.1.1.1. Steal user(s) ID and password(s) (OR)	I1
1.1.1.1.1.1.2. Steal private key(s) and password(s) (OR)	I1
1.1.1.1.1.1.3. Steal hardware token(s) and password(s) or PINS	I1
1.1.1.1.1.1.4. Recruit help of trusted insider (OR)	
1.1.1.1.1.1.4.1. Plant spy as trusted insider (OR)	
1.1.1.1.1.1.4.2. Compromise trusted insider	
1.1.1.1.1.1.4.2.1. Bribe (OR)	
1.1.1.1.1.1.4.2.2. Threaten (OR)	
1.1.1.1.1.1.4.2.3. Blackmail (OR)	
1.1.1.1.1.1.4.2.4. Fool	
1.1.1.1.1.1.5. Exploit flaws in key management/distribution system (OR)	I1
1.1.1.1.1.1.6. Get information through computer's Internet connection (OR)	I1
1.1.1.1.1.1.6.1. Monitor communications over Internet for leakage (OR)	I1
1.1.1.1.1.1.6.2. Use virus to send information over Internet	I1
1.1.1.1.1.1.7. Get information through computer's public telephone network connection (OR)	I1
1.1.1.1.1.1.7.1. Tap PTN and monitor for leakage of sensitive information	I1
1.1.1.1.1.1.8. Physically scavenge discarded items (OR)	I1
1.1.1.1.1.1.8.1. Inspect dumpster content on-site (OR)	I1
1.1.1.1.1.1.8.2. Inspect refuse after removal from site	I1
1.1.1.1.1.1.9. Monitor electronic emissions from computer or network (OR)	I1
1.1.1.1.1.1.9.1. Determine optimal physical monitoring position (AND)	I1
1.1.1.1.1.1.9.2. Acquire necessary monitoring equipment (AND)	I1
1.1.1.1.1.1.9.3. Setup monitoring site (AND)	I1
1.1.1.1.1.1.9.4. Analyze emissions from site	I1
1.1.1.1.1.2. Physically access computer (OR)	
1.1.1.1.1.2.1. Break into building and/or computer room	
1.1.1.1.1.2.1.1. Pick lock (OR)	
1.1.1.1.1.2.1.2. Steal key (OR)	
1.1.1.1.1.2.1.3. Forge key (OR)	

ATTACK COUNTERMEASURE

1.1.1.1.1.2.1.4. Recruit help of trusted insider (1.1.1.1.1.1.4)
1.1.1.1.1.3. Access computer through intranet (OR)
1.1.1.1.1.3.1. Gain access to another computer on intranet (1.1.1.1.1) I1
1.1.1.1.1.4. Access computer through Internet (OR)
1.1.1.1.1.4.1. Gain privileged access to Web server
1.1.1.1.1.4.1.1. Identify domain name (AND)
1.1.1.1.1.4.1.2. Identify firewall IP address (AND)
1.1.1.1.1.4.1.2.1. Interrogate domain name server (AND)
1.1.1.1.1.4.1.2.2. Scan for firewall identification (AND)
1.1.1.1.1.4.1.2.3. Trace route through firewall to web server
1.1.1.1.1.4.1.3. Determine firewall access control (AND)
1.1.1.1.1.4.1.3.1. Search for specific default listening ports (AND)
1.1.1.1.1.4.1.3.2. Scan ports broadly for any listening port
1.1.1.1.1.4.1.4. Identify web server operating system and type (AND)
1.1.1.1.1.4.1.4.1. Scan OS services' banners for OS identification (AND)
1.1.1.1.1.4.1.4.2. Probe TCP/IP stack for OS characteristic information
1.1.1.1.1.4.1.5. Exploit web server vulnerabilities
1.1.1.1.1.4.1.5.1. Access sensitive shared intranet resources directly (AND)
1.1.1.1.1.4.1.5.2. Access sensitive data from privileged account on web server
1.1.1.1.1.5. Access computer using public telephone network (PTN)
1.1.1.1.1.5.1. Obtain dialup access number (1.1.1.1.1.1)
1.1.1.1.2. Modify software (AND) V1 V2
1.1.1.1.2.1. Modify source code (OR) V1 V2
1.1.1.1.2.2. Modify object code (OR) V1 V2
1.1.1.1.2.3. Modify compiler to emit rogue object code V1 V2
1.1.1.1.3. Fool software certification process V1 V2
1.1.1.1.3.1. "Hide" rogue software change in convoluted source code logic (OR) V1 V2
1.1.1.1.3.2. Modify the compiler to emit rogue object code (1.1.1.1.2.3) (OR) V1 V2
1.1.1.1.3.3. Recruit help of certification tester or official (1.1.1.1.1.1.4) V1 V2
1.1.1.2. Modify object code after it is loaded into target computer (OR) V1 V2 I1
1.1.1.2.1. Replace object code storage device (OR) V1 V2
1.1.1.2.1.1. Gain physical access to loaded computer (AND)
1.1.1.2.1.1.1. Break into vendor's storage facility (1.1.1.1.1.2.1) (OR)
1.1.1.2.1.1.2. Break into vehicle during transport of computer to jurisdiction (1.1.1.1.1.2.1) (OR)
1.1.1.2.1.1.3. Break into election management center (1.1.1.1.1.2.1) (OR)
1.1.1.2.1.1.4. Break into jurisdiction warehouse (1.1.1.1.1.2.1) (OR)
1.1.1.2.1.1.5. Break into vehicle during transport of computer to poll site (1.1.1.1.1.2.1) (OR)
1.1.1.2.1.1.6. Break into poll site (OR) (1.1.1.1.1.2.1)
1.1.1.2.1.1.7. Physically access computer during election

ATTACK	COUNTERMEASURE							
1.1.1.2.1.1.7.1. Access DRE while voting (OR)								
1.1.1.2.1.1.7.2. Recruit poll worker to allow access to DRE (1.1.1.1.1.1.4) (OR)								
1.1.1.2.1.1.7.3. Recruit poll worker to allow access to poll worker computer (1.1.1.1.1.1.4) (OR)								
1.1.1.2.1.1.7.4. Recruit jurisdiction worker to allow access to election management computer (1.1.1.1.1.1.4)								
1.1.1.2.1.2. Replace object code storage device	V1	V2						
1.1.1.2.1.2.1. Replace EEPROMS (OR)	V1	V2						
1.1.1.2.1.2.2. Replace hard disks or floppy disks	V1	V2						
1.1.1.2.2. Change stored object code or data (OR)	V1	V2		I1				
1.1.1.2.2.1. Gain physical access to loaded computer (1.1.1.2.1.1) (AND)								
1.1.1.2.2.2. Obtain privileged access information for user or administrator account (1.1.1.1.1.1) (AND)				I1				
1.1.1.2.2.3. Rewrite hard disk or floppy disks (OR)	V1	V2						
1.1.1.2.3. Change running object code or data	V1	V2		I1				
1.1.1.2.3.1. Physically access computer during election (1.1.1.2.3.1) (AND)								
1.1.1.2.3.2. Obtain privileged access information for user or administrator account (1.1.1.1.1.1) (AND)				I1				
1.1.1.2.3.3. Patch main memory during election	V1	V2						
1.1.1.3. Modify election configuration after it is loaded into DRE or election management computer	V2	V3	I1	I2	I3			
1.1.1.3.1. Modify software to change election configuration (1.1.1) (OR)				I1				
1.1.1.3.2. Get access to loaded computer				I1				
1.1.1.3.2.1. Gain physical access to loaded computer (1.1.1.2.1.1) (AND)								
1.1.1.3.2.2. Obtain privileged access information for user or administrator account (1.1.1.1.1.1)				I1				
1.1.1.3.3. Change election configuration (AND)	V2	V3						
1.1.1.3.3.1. Change ballot definition (OR)	V2.1							
1.1.1.3.3.1.1. Change on-screen ballot presentation to fool voter (OR)	V2.1							
1.1.1.3.3.1.2. Substitute different signed blank ballot to fool voter	V3.2	I1						
1.1.1.3.3.1.2.1. Steal LEO key (1.1.1.1.1.1) (AND)				I1				
1.1.1.3.3.1.2.2. Change correspondence between answer marks and answers (AND)	V3.2							
1.1.1.3.3.1.2.3. Generate new signed blank ballot								
1.1.1.3.3.2. Change codebook key to generate false codebook correspondence (OR)	V2	V3.6						
1.1.1.3.3.3. Change published codebook commitments or election parameters	V2	V3.6						
1.1.1.3.4. Rewrite hard disk or floppy disks (OR)								
1.1.1.3.5. Patch main memory during election								
1.1.2. Fool L&A testing	V2							
1.1.2.1. Modified software acts honestly for pre-election and post-election L&A test (AND)	V2							
1.1.2.2. Modified computer activates rogue “payload” for actual election	V2							
1.2. Change ballot box contents (OR)	V1	V2	V3	I1	I2	I3	P1	P2
1.2.1. Change voted ballot (OR)	V1	V2	V3	I1	I2	I3		
1.2.1.1. Capture other than voter intent (1.1) (OR)	V1	V2	V3	I1	I2	I3		
1.2.1.2. Modify ballots in ballot box	V1			I1				

ATTACK — COUNTERMEASURE

1.2.1.2.1. Modify ballot storage device in poll site (OR) V1
1.2.1.2.1.1. Change software to re-write ballot storage device (1.1.1.2) (OR) V1 I1
1.2.1.2.1.2. Replace ballot storage device (OR) V1
1.2.1.2.1.3. Remove and rewrite ballot storage device V1 I1
1.2.1.2.2. Modify ballot storage device during transport to tabulation center (OR) V1
1.2.1.2.2.1. Replace ballot storage device (OR) V1
1.2.1.2.2.2. Rewrite ballot storage device V1 I1
1.2.1.2.3. Modify ballot storage device at tabulation center (1.2.1.2.2)(OR) V1 I1
1.2.1.2.4. Change ballots during transmission to election management computer (OR) V1
1.2.1.2.4.1. Tap PTN and modify data stream (OR) V1
1.2.1.2.4.2. Attack Internet connection and modify data stream (1.1.1.1.1.4) V1
1.2.1.2.5. Change ballots after loading into election management computer V1 V3 I1
1.2.1.2.5.1. Modify election management software or configuration (1.1.1) (AND) V1 V3 I1
1.2.1.2.5.2. Change ballots in election management computer V1 V3
1.2.1.2.5.2.1. Change ballots before Neff Shuffle (OR) V1 V3
1.2.1.2.5.2.2. Change ballots during Neff Shuffle (OR) V1 V3
1.2.1.2.5.2.3. Change ballots after Neff Shuffle (OR) V1 V3

1.2.2. Add ballot (OR) I1 I2 I3 P1 P2
1.2.2.1. Use voting ticket to vote multiple times (OR) I2
1.2.2.2. Use DRE to vote using illegally obtained voting ticket (OR) I1 I2 I3 P1 P2
1.2.2.2.1. Alter voter roll (AND) P2
1.2.2.2.1.1. Add name (e.g. dead person) to voter roll (OR) P2
1.2.2.2.1.1.1. Modify jurisdiction's voter registration database (OR) I1 P2
1.2.2.2.1.1.1.1. Modify election configuration after it is loaded into target computer (1.1.1.3) I1
1.2.2.2.1.1.2. Modify poll worker's computer (OR) I1 P2
1.2.2.2.1.1.2.1. Modify election configuration after it is loaded into target computer (1.1.1.3) I1
1.2.2.2.1.1.3. Modify printed voter rolls P2
1.2.2.2.1.1.3.1. Break into jurisdiction facility (1.1.1.1.1.2.1) (OR)
1.2.2.2.1.1.3.2. Intercept voter rolls during transport to poll site (OR)
1.2.2.2.1.1.3.3. Recruit help of jurisdiction worker before election (1.1.1.1.1.1.4) (OR)
1.2.2.2.1.1.3.4. Recruit help of poll worker during election
1.2.2.2.1.2. Identify registered non-voter
1.2.2.2.1.2.1.1. Monitor voters entering poll site
1.2.2.2.2. Obtain voting ticket (AND)
1.2.2.2.2.1. Provide false identification to poll worker for added voter or non-voter (OR) P1 P2
1.2.2.2.2.2. Recruit help of poll worker during election (1.1.1.1.1.1.4) (OR) P1 P2
1.2.2.2.2.3. Forge voting ticket (AND) I1 I2 I3

ATTACK COUNTERMEASURE

1.2.2.2.2.3.1. Compromise poll worker's voter authorization private key I1 I2 I3
1.2.2.2.2.3.1.1. Steal poll worker's voter authorization private key (OR) I1
1.2.2.2.2.3.1.2. Forge poll worker's voter authorization private key (OR) I2 I3
1.2.2.2.2.3.1.3. Recruit help of jurisdiction worker (1.1.1.1.1.1.4) (OR)
1.2.2.2.2.3.1.4. Recruit help of poll worker (1.1.1.1.1.1.4)
1.2.2.2.3. Falsify voter roll to check off voter P1 P2
1.2.2.2.3.1. Recruit help of poll worker during election (1.1.1.1.1.1.4) (OR)
1.2.2.2.3.2. Recruit help of jurisdiction worker after election (1.1.1.1.1.1.4) (OR)
1.2.2.2.3.3. Break into jurisdiction facility (1.1.1.1.1.2.1) (OR)
1.2.2.2.3.4. Break into election management (1.1.1)
1.2.2.3. Generate ballot and add to ballot box V3 I1 I2 I3 I4 P1 P2
1.2.2.3.1. Alter voter roll (1.2.2.2.1) (AND) P2
1.2.2.3.2. Forge voting ticket (1.2.2.2.2.3) (AND) I1 I2 I3
1.2.2.3.3. Select valid, unique BSN from codebook commitments (AND) V3 I4
1.2.2.3.4. Generate encrypted voted ballot using BSN, signed voting ticket and published election parameters (AND)
1.2.2.3.5. Steal DRE private key (AND) I1
1.2.2.3.5.1. Get access to election management computer (1.1.1) (OR) I1
1.2.2.3.5.2. Steal configuration from distribution medium (OR) I1
1.2.2.3.5.2.1. Break into facility or vehicle (1.1.1.1.1.2.1) (OR)
1.2.2.3.5.2.2. Recruit help of jurisdiction worker (1.1.1.1.1.1.4) (OR)
1.2.2.3.5.2.3. Recruit help of poll worker (1.1.1.1.1.1.4)
1.2.2.3.5.3. Get access to DRE after configuration is loaded (1.1.1.2.1) I1
1.2.2.3.6. Sign encrypted voted ballot with DRE private key (AND)
1.2.2.3.7. Insert ballot in ballot box (1.2.1.2) (AND) P2
1.2.2.3.8. Falsify voter roll to check off voter (1.2.2.2.3) P1 P2
1.2.2.4. Tabulate before the polls close and target "get out the vote" V3.1
1.2.2.4.1. Get access to ballot box before polls close (AND)
1.2.2.4.2. Tabulate ballot box
1.2.3. Delete ballot (OR) V1 I1
1.2.3.1. Modify DRE software (1.1.1) (OR) I1
1.2.3.1.1. Change DRE software to discard ballots after verification receipt printed I1
1.2.3.2. Modify ballots in ballot box (1.2.1.2) I1
1.2.3.2.1. Delete ballot from ballot box I1
1.2.3.3. Assert voter forged the verification receipt if discrepancy produced V1
1.2.4. Delete ballot and add different ballot (same as Delete (1.2.3) plus Add (1.2.2) or Change (1.2.1))
1.3. Change election results V1 V2 V3 I1 I2 I3 P1 P2
1.3.1. Change encrypted ballot box contents (OR) V1 V2 V3 I1 I2 I3 P1 P2
1.3.1.1. Modify election management software, data storage or output (1.1.1) (AND) I1

ATTACK	COUNTERMEASURE
1.3.1.2. Substitute different encrypted ballots (1.2.4)	V1 V2 V3 I1 I2 I3 P1 P2
1.3.2. Change decrypted ballots (OR)	V3.5 I1
1.3.2.1. Modify election management software, data storage or output (1.1.1)	I1
1.3.2.2. Substitute different decrypted ballots	V3.5
1.3.3. Publish false tally and false election transcript	V1 V2 V3 I1
1.3.3.1. Steal trustee private key (AND)	V3.1
1.3.3.1.1. Break key sharing protocol (OR)	V3.1
1.3.3.1.2. Recruit threshold number of trustees to give up their key shares (1.1.1.1.1.4)	
1.3.3.2. Steal LEO key (1.1.1.1.1) (AND)	I1
1.3.3.3. Generate and sign false election transcript (AND)	
1.3.3.4. Modify election management software, data storage or output (1.1.1)	I1
1.3.3.4.1. Publish false tally and false election transcript	V1 V2 V3 (V3.4 V3.5 V3.6 V3.7)
2. Goal: Compromise voter privacy	V3 I1
2.1. Determine voter's selections from published verification statement (OR)	
2.1.1. Match voter with published verification statement (AND)	I1
2.1.1.1. Steal voter's verification receipt (OR)	
2.1.1.2. Match voting ticket number with published verification statement (OR)	
2.1.1.2.1. Recruit poll worker to record voter's voting ticket number (1.1.1.1.1.1.4)	
2.1.1.3. Match voter order with order of published verification statements	I1
2.1.1.3.1. Record voter names in order that they voted (AND)	
2.1.1.3.1.1. Visually monitor voting in poll site (OR)	
2.1.1.3.1.2. Recruit help of poll worker to make list as voters are authenticated and use DRE (1.1.1.1.1.1.4)	
2.1.1.3.2. Defeat random storage of ballots	
2.1.1.3.2.1. Modify DRE software (1.1.1) (OR)	I1
2.1.1.3.2.2. Modify election management software (1.1.1)	I1
2.1.1.3.2.3. Fool L&A testing (1.1.2)	
2.1.2. Generate codebook for BSN in verification statement	
2.1.2.1. Steal codebook key (AND)	V3 V3.1.1 I1
2.1.2.1.1. Recruit threshold number of trustees to give up their key shares (1.1.1.1.1.1.4) (OR)	V3.1
2.1.2.1.2. Get access to election management computer (1.1.1) (OR)	I1
2.1.2.1.3. Steal configuration from distribution medium (OR)	I1
2.1.2.1.3.1. Break into facility or vehicle (1.1.1.1.1.2.1) (OR)	
2.1.2.1.3.2. Recruit help of jurisdiction worker (1.1.1.1.1.1.4) (OR)	
2.1.2.1.3.3. Recruit help of poll worker (1.1.1.1.1.1.4)	
2.1.2.1.4. Get access to DRE after configuration is loaded (1.1.1.2.1)	I1
2.2. Modify DRE software to capture and record voter's selections (OR)	I1
2.2.1. Modify DRE software to record unencrypted voter selections (1.1.1) (AND)	I1
2.2.2. Record voter names in order that they voted (2.1.1.3.1)	

ATTACK COUNTERMEASURE

2.2.3. Retrieve ballot choices from DRE during or after election (1.1.1.2.1) I1
2.3. Determine voter's selection from encrypted ballot in published election transcript (OR) I1
2.3.1. Match voter order with order of stored ballots (2.1.1.3) (AND) I1
2.3.2. Recruit threshold number of trustees to decrypting ballot or give up their key share (1.1.1.1.1.1.4) (OR) V3.1
2.3.3. Break public-key encryption (OR) I1
2.3.3.1. Bias the Random Number Generator in the DRE (OR) I1
2.3.3.2. Brute force key (OR) I1
2.3.3.3. Discover weakness in algorithm I1
2.3.4. Break the Neff Shuffle protocol I1
2.3.4.1. Discover flaw (e.g., generate a proof without proper shuffle) (OR) V3 I1
2.3.4.2. Bias the Random Number Generator used by the trustees (OR) I1
2.3.4.3. Recruit threshold number of trustees to decrypt ballot or give up key shares (1.1.1.1.1.1.4) V3.1
2.4. Use hidden camera in poll site to record voter's choices P8
2.4.1. Install one or more very small cameras in public poll site well before election (AND)
2.4.2. Focus each camera to take shot of voter's face and DRE screen (AND)
2.4.3. Recover camera(s) from public poll site after election

3. Goal: Buy vote or coerce voter V3 I1 P4 P5 P6
3.1. Recruit voter to cast vote a certain way (1.1.1.1.1.1.4) (AND)
3.2. Prove how voter voted V3 I1 P4 P5 P6
3.2.1. Use hidden camera in poll site (2.4) (OR)
3.2.2. Use verification statement (OR)
3.2.2.1. Recruit voter to surrender verification receipt (1.1.1.1.1.1.4) (AND) P4
3.2.2.2. Get BSN from verification receipt (AND)
3.2.2.3. Use BSN to retrieve matching published verification statement (AND)
3.2.2.4. Capture codebook for BSN V3 I1 P5 P6
3.2.2.4.1. Steal codebook key (2.1.2.1) (OR) V3 I1
3.2.2.4.2. Recruit voter to capture codebook from DRE P5 P6
3.2.2.4.2.1. Give voter concealable recording device (AND)
3.2.2.4.2.1.1. Give voter serial printer identical to observer printer (OR)
3.2.2.4.2.1.2. Give voter digital camera (e.g., cell phone camera)
3.2.2.4.2.2. Voter smuggles recording device into polling station (AND)
3.2.2.4.2.3. Voter records codebook P5 P6
3.2.2.4.2.3.1. Voter prints codebook (OR) P5
3.2.2.4.2.3.2. Voter takes pictures of on-screen codebook P6
3.2.3. Use encrypted ballot I1
3.2.3.1. Capture configuration parameters on verification receipt (OR)
3.2.3.1.1. Modify DRE to "leak" configuration parameters to voter (1.1.1) (AND) I1

ATTACK COUNTERMEASURE

3.2.3.1.1.1. Print election parameters on verification receipt (OR)
3.2.3.1.1.2. Print codebook on verification receipt (OR)
3.2.3.1.1.3. Print codebook key on verification receipt
3.2.3.1.2. Use configuration parameters to determine how voter voted
3.2.3.1.2.1. Decrypt encrypted ballot with matching BSN in election transcript (OR)
3.2.3.1.2.2. Use codebook to determine votes from verification statement (OR)
3.2.3.1.2.3. Generate codebook from codebook key and use codebook to determine votes from verification statement
3.2.3.2. Modify DRE to generate election parameters with a known pseudo-random process (1.1.1) I1
3.2.3.2.1. Decrypt encrypted ballot with matching BSN in election transcript