Applying a Reusable Election Threat Model at the County Level

Eric L. Lazarus[1], David L. Dill[2], Jeremy Epstein[3], and Joseph Lorenzo Hall[4,5]

[1]DecisionSmith
[2]Stanford University; Computer Science Department
[3]SRI International; Computer Science Laboratory
[4]University of California, Berkeley; School of Information
[5]Princeton University; Center for Information Technology Policy

Abstract

We describe the first systematic, quantitative threat evaluation in a local election jurisdiction in the U.S., Marin County, California, in the November 2010 general election. We made use of a reusable threat model that we have developed over several years. The threat model is based on attack trees with several novel enhancements to promote model reuse and flexible metrics, implemented in a software tool, AttackDog. We assess the practicality of reusable threat models for local elections offices and analyze specific vulnerabilities in Marin County, using as our metric "attack team size" (ATS) – the number of individuals who are knowingly involved in election fraud.

1 Introduction

Democracy would be easy and elections would be simple if people always were in agreement. But the most vital function of an election is to serve as a decision-making mechanism that is respected by the losing parties, even when choices are contentious. These are also the circumstances in which suspicion flourishes, and the results of elections will not necessarily be trusted unless they are *evidently* accurate. In particular, if there is a likelihood that intent of the voters could be misrepresented in the outcome of the election, without being detected, that outcome may not be respected by the public, undermining the foundations of democratic governance.

Since the extended uncertainty in the November 2000 Presidential election, caused in part by the inaccuracy of punched card ballots, there has been a great deal of focus on election technology in the U.S. The subsequent widespread introduction of touch-screen machines (more accurately, *direct-recording electronic*, or *DRE*, voting machines) raised concerns about the potential for undetected error or fraud enabled by complex and opaque technology. The primary value of computer expertise in election policy has been to demonstrate the insecurity of numerous electronic voting systems, and to recognize and communicate the futility of trying to secure computer systems through purely electronic means. Instead, many systems in the U.S. rely on post-election auditing, where paper ballots (filled out by the voter by hand or by machine) are chosen at random for hand-counting to check electronic totals.

There is a pressing problem that is receiving far too little attention: how to discover and address vulnerabilities in the election system as a whole, including procedures, not just technology. Since election administration is conducted primarily at the local level in the U.S.,[1] the only solution to this problem that we see is *universal, systematic threat evaluation of election systems in local jurisdictions*.

One obvious barrier to universal threat evaluation is that it requires a great deal of effort and expertise. Our proposed solution to this problem is to use a *reusable threat model*, which can be applied to a local jurisdiction relatively easily. Since election procedures are similar across U.S. jurisdictions, even in different states, reusable models could avoid wasteful duplication of effort, greatly reducing the cost of evaluating a particular jurisdiction. Perhaps more importantly, a reusable threat model would provide a means to share knowledge about best security practices across many jurisdictions. When one jurisdiction is shown to be less secure than another, the model would show the different options and assumptions that explain the differences.

[1]Depending on the location in the U.S., details of election administration are usually managed at the county or city level. In some parts of the U.S., cities within counties have their own election systems, and the same voter may vote in elections conducted by the city or county at different times. So the concept of "jurisdiction" can be a bit complicated.

Our reusable model was based on *attack trees*. An attack tree explicitly captures the goal of the attackers, the individual steps to achieving those goals, and the defenses against those attacks. In addition, specific costs are associated with attacks, so that they can be compared quantitatively. This effort was supported by a software tool we developed, AttackDog, that enabled the definition, editing, and evaluation of attack trees. More importantly, the attack trees in AttackDog have several novel enhancements that support the development of *reusable* threat models, by parameterizing attack trees and their associated cost functions.

The basics of voting in Marin County

To understand the rest of the paper, it is necessary to know a little about how elections work in Marin County. For polling place voting, Marin County uses Diebold[2] AccuVote-OS precinct-count optical scan systems and AutoMARK ballot marking devices for accessible voting. Voters mark ballots by hand filling out bubbles, and insert them into the scanner at the polling place, which counts votes on the ballots and stores the ballots in an attached ballot box. Like other counties in California, Marin County receives a significant fraction of its ballots through the mail.[3] In VbM, each voter receives a ballot and a numbered envelope. Mail ballots can be returned through the U.S. Postal Service or delivered in person to the elections office or a polling place on election day.

California law requires a manual count of the paper ballots in at least 1% of the precincts (we call these "manual audits" in the rest of the paper), chosen at random, and these numbers are compared to the count from the scanners in the polling place.

Overview of the rest of the paper

In the remainder of the paper, we discuss our methodology for developing and applying reusable threat models. We describe our approach to attack trees and how we worked to use them in practice in Marin County in Section 2. In this study, the measure of vulnerability we use is "attack team size" (ATS) – the number of individuals who are knowingly involved in the attack. We argue that elections are more vulnerable if a small attack team can change the result of the election with a low probability of the fraud being detected. Interestingly, the use of a quantitative metric immediately focuses attention on procedures and away from "hardening" of computer system security, because existing computer security practices do not, in general, result in increased attack team size (particularly when insiders can be attackers). In Section 3, we outline three specific attacks relevant to Marin County with small ATS, and also the results of an alternative metric calculation that distinguishes insiders versus outsiders on an attack team. We discuss related work in Section 4 and end with a discussion of the current work, its limitations and future directions in Section 5.

2 Reusable threat models

There are no perfectly secure systems. With sufficient resources and a sufficient tolerance for risk, virtually any system can be attacked successfully. Therefore, an analysis should answer the question: How should limited resources be deployed to make the system as trustworthy as possible? Answering this question requires a *quantitative*, *comparative* approach to threat evaluation.

2.1 Attack trees

Election processes and security issues are complex, and discussion rapidly becomes confusing. A structured approach to threat evaluation has the advantage of separating the various issues so that they can be considered in an organized way.

We have chosen a generalized form of *attack trees* as our structure for threat analysis. An attack tree is essentially a tree of AND nodes, OR nodes, and leaves. The top node in the tree represents the attacker's goal (e.g., "Change Result of Election Successfully" in our tree).[4]

Children of a node represent subgoals, methods or categories of attacks. AND nodes represent multiple goals that must be achieved in order to achieve the parent goal. OR nodes represent alternative ways to achieve the parent goal. Leaves of the tree represent individual steps of an attack.

An *attack* is a collection of leaf nodes. Intuitively, an attack is something like a plan for achieving the top-level goal, although the steps are not specified in sequence. An attack *satisfies* a leaf node if the leaf appears in the attack; an AND node is satisfied if all of its children nodes are satisfied; and an OR node is satisfied if at least one of its children is satisfied. A *successful attack* is an attack that satisfies the top-level node in the tree (the attacker's primary goal). Note that an attack tree can have many successful attacks.

[2] now rebranded as Premier Election Solutions.

[3] California has few restrictions on vote-by-mail, and allows voters to designate themselves as "permanent absentee voters," who automatically receive mail ballots in every election.

[4] AttackDog is based on two-player game. It would be an interesting problem to extend attack trees to deal with more than two players to model different attackers with competing goals.

Attacks can be quantified by computing *cost* for each attack. For example, the metric can be considered a "cost" if a lower number represents attacks that are more attractive to the attacker and more dangerous to the defender. A cost could be monetary, or something else (the metric we actually used in the study, described below, is not monetary). The metric could also be a random variable (with a probability distribution), although it is not in this study. Cost could also be a risk.

The cost of an attack is computed by associating a collection of numerical and non-numerical attributes with each step, and providing a combining function to compute the attack cost from the attribute values of the individual steps. This function could be something as simple as taking the sum or maximum over the steps of the attack, or it could be more complex. *An important difference between our generalized attack trees and previous definitions is that the costs are not computed by recursive traversal of the tree.* The attacks (lists of steps) are generated by recursive traversal, and a cost is computed on each individual attack. This approach has higher computational cost (which is still trivial) and is much more flexible than computing costs directly by recursive traversal.

For this project, we used our software tool, AttackDog, to assist in defining and evaluating attack trees. AttackDog provides menus for defining and annotating nodes in an attack tree. It allows the user to associate one or more attributes with the leaf nodes, for use in computing costs, and allows the user to provide an arbitrary function for computing the cost of entire attacks. It then lists all of the attacks and their costs, which can be exported to a spreadsheet. Each attack is a list of attack steps that must be performed (not necessarily in chronological order) and an overall cost for the attack (ATS in this study). Interpretation of the results requires examining the generated attacks along with the original tree from which they were generated. Although many attacks can be appear in the attack list, they are easily sorted so that the attacks with minimum ATS appear at the beginning of the list, so that these can be detected in detail. User often want to look up attacks with a particular step to understand why it does *not* have a small ATS, which can easily be done be searching the attacks. For this project, we also selected individual subtrees of the top-level OR node by using "omit" nodes to suppress the others, so we could inspect the attacks for each subtree individually.

Since defining a new attack tree is labor intensive, AttackDog has several features to facilitate the reuse of threat models. Arbitrary parameters can be defined separately, and the leaf attributes can depend on these parameters. A generic attack tree can be tailored to individual jurisdictions by setting parameters appropriately. For example, there is a parameter for the number of precincts in a jurisdiction. There is also a special "omit" attribute for each node in the tree, the value of which depends on other parameters. If the "omit" attribute for a node is true, the tree is analyzed as though the node were not in the tree. The omit attribute can be used to tailor trees to consider different scenarios, such as how the presence or absence of security measures affects attack costs. It can be used, for example, to remove an entire subtree in jurisdictions that lack a particular defensive measure or are immune to a class of attacks. For example, an important defense against malware or other computer-based tampering with election results is post-election manual auditing of ballots in randomly selected precincts to verify electronic totals. In jurisdictions using DREs that do not produce auditable paper ballots, the entire audit subtree would be omitted.

It is important to have appropriate expectations of AttackDog. It is a software tool for *supporting* threat evaluation. It can help organize the analysis and generates attacks with approximate costs. Obviously, it will not "discover" attacks unless the steps are spelled out, and the accuracy of the results rests on the accuracy of the assumptions in the model. The primary sources of errors and disagreements about threat analyses are the assumptions, not computations. AttackDog can help document these assumptions and trace their effects on the costs of the generated attacks, but it cannot ensure the correctness of the assumptions.

2.2 Metric: attack team size

In AttackDog, the cost of an attack can be computed in any way and with whatever inputs the author of the threat model wishes. It provides the mechanism for defining attack costs but not the policy. For a particular threat evaluation, it is necessary to make the policy decision about how to quantify threats.

Quantifying threats is a task that should be undertaken with great care, because there will obviously be great uncertainty in estimating the various parameters, and that uncertainty itself is difficult to estimate. There is little empirical data concerning election fraud that would be useful for quantifying costs (and, by definition, no data on *undetected* election fraud). We believe that the best that can be done, for now, is to choose a very simple metric which is robust to order-of-magnitude errors in parameters, and then agree on reasonable "best guesses" for those parameters.

Some obvious metrics, such as money, seem to be flawed. Given the economic value of control of the government, there would seem to be ample financial

resources available for election fraud, as witnessed by the escalating costs of campaigns, even sometimes for local offices. For example, even in large counties, elected local officials make high-stakes decisions about contracting and zoning. Influence over decisions by Federal-level offices can be vastly more valuable. Risk of detection and punishment is likely to be a more important consideration for a group considering committing election fraud than financial cost.

Based on these considerations, we have chosen to use ATS as our primary metric for election security. The attack team is the set of people who knowingly participate in election fraud. The most dangerous attacks are those with small ATS. While ATS may not be the best metric for some kinds of threat evaluations, we believe it is appropriate for U.S. elections, where the potential benefits of election theft are huge, and the primary deterrent is the detection of attempted or actual fraud. Detection is undesirable for attackers because it (probably) entails significant legal penalties as well as thwarting the attackers' goals.

In this context, there are several justifications for ATS as a metric. A larger ATS greatly increases the risk of exposure if a team member is caught "in the act," or boasts or confesses of his participation. There is also a substantial risk that recruiting team members will result in exposure or the infiltration of the attack team by people who could expose the fraud. Secondarily, ATS scales approximately with resource requirements for attacks, such as monetary cost, person-hours, etc., so it results will be roughly consistent with other metrics that might be proposed. So far as we know, ATS was first proposed as a metric by the first author when he led a more informal study of the comparative vulnerabilities of different technology. [15] In that study, ATS was called "number of informed participants."

We exploit the flexibility of AttackDog's attack cost computations to account for the possibility that a single attack team member may be able to perform several steps. Attackers are categorized as *election insiders* (trusted election or voting equipment vendor staff), *poll workers, audit workers* (people performing a hand count of paper ballots, if such a process is used), *postal insiders*, and *outsider attackers* (anyone not in the previous categories). Each step requires a certain number of people in one or more categories. However, the same member of the category can perform multiple steps. So, to compute the ATS for an attack, the maximum number of people from each category required to perform any step in the attack is computed, and the ATS is the total number of people in all categories. For example, if an attack requires ten election insiders for a particular step, and another step requires five election insiders, it is assumed that the ten election insiders who did the first step are available to do the second step, so the total number of election insiders is ten, not fifteen. For this study, it is not necessary to consider individuals who could belong to several categories (e.g., insiders vs. poll workers) because there is little overlap in practice.

A criticism of ATS is that different classes of individuals are more valuable to an attack team, or more difficult to recruit, and that members of these classes should be assigned different values or costs. Unfortunately, assigning weights to different classes of attackers adds more parameters to the model that have to be justified. The best way to address this is to test the results of the analysis for sensitivity to such considerations, as we do in the next section. Importantly, the analysis methodology can accommodate many different types of metrics, including very complex ones, making the consideration of alternative metrics feasible.

2.3 The reusable election threat model

Our proposed approach to threat modeling in local jurisdictions is to define, maintain, and evolve a formal reusable threat model for *all* local jurisdictions. The model consists of two separate parts: A jurisdiction-independent parameterized attack tree, and a set of parameters (variable definitions) for the particular locality. Parameters can be quantitative or qualitative. Both types of parameters can appear in arbitrary formulas in a high-level programming language (the "R" language in AttackDog[5]) which can be used to compute the costs of attacks.

The "omit" nodes mentioned previously, are a simple but important feature that allows a single tree to be reused for many jurisdictions (among other uses). Boolean formulas using parameters can implicitly remove inapplicable parts of the tree from consideration, using the "omit" node feature, without requiring actual changes to the tree structure. The parts of the tree that don't apply in a jurisdiction can simply be disabled.

Prior to this study, we had developed an extensive election threat model over a period of several years. The initial tree was based on a broad threat analysis of elections conducted as part of an investigation conducted with the Brennan Center for Justice in 2006. [15] Since that time, the tree has been repeatedly extended, refined, and reorganized by several different people. Detailed threats that came from examining local jurisdictions including in Leon County, Florida, and more attacks were added based on a threat-analysis prepared by the Election Assistance Commission [5],

[5] See: `http://www.r-project.org/`

substantial portions of which, in turn, were based on an earlier threat model of ours. The threat model is parameterized to allow it to model different situations and jurisdictions.

To bound the scope of the study, we chose to focus on defenses against malicious attacks on elections that were the most likely to lead to undetected changes in the election outcome. We also limited our consideration to election administration, excluding issues such as the conduct of campaigns. We recognize that this scope does not include all interesting election security questions. Attackers might have other, or additional, goals in an election. For example, attackers might wish to disenfranchise, mislead, or intimidate voters. These are attacks that are probably detectable (there would be numerous witness and victims that might complain), but perhaps deniable ("it was an honest mistake") and possibly not correctable. Indeed, many recent election controversies have centered on charges of such attacks, and these disputes were often not resolved. In other work, we have developed attack trees that include some attacks of these types, but not included them in the attack trees for this study. This limitation is scope is to bound the size of an already difficult problem. For example, the ATS metric limits the risk for attackers who wish to avoid detection, but is it the best metric for attacks that will be detected anyway? Perhaps the best approach to more comprehensive threat evaluation would be to consider different kinds of goals, possibly with different vulnerability metrics.

In our reusable tree, the top-level node of the tree describes the attacker's goal: to change the election. All attacks apply to DREs or optical scan systems. It is an "OR" node, with nodes for several alternative types of under it:

1. Change Result of Election Successfully
    - 1.1 Attack Voting Equipment
    - 1.2 Pollworker Attack
    - 1.3 Perform Voter Impersonation Attack
    - 1.4 Perform Vote By Mail Attack

Each of the second-level nodes is the root of a tree. Many of these trees are extensive, with 100 or more nodes, and pages of textual explanation. For example, the "Attack Voting Terminal" node reflects the long debate over the past few years about whether computerized voting equipment can be trusted and a myriad of methods for subverting voting equipment. "Attack Voting Equipment" (making the software or hardware behave maliciously) In turn, "Attack Voting Equipment" is an AND node, since the attacks involve a series of steps that have to be executed successfully. An attacker must

1. Gather Needed Technical Knowledge (e.g., learn enough about the machine to hack it);
2. Develop and Insert Malware or Misconfiguration (e.g. insert a virus).
3. Get Through Inspection (disguise the effects of hacking the machine from anyone studying the machines);
4. Get Through Pre-Election Testing (make sure that the machines do not cheat during testing);[6]
5. Render Routine Statistical Audit Ineffective (defeat efforts to double-check the results of the machine by hand-counting paper ballots in random precincts). A jurisdiction-specific parameter omits this node if there *are* no audits (e.g., if there are no paper ballots), eliminating the need for the attacker to deal with that subgoal.

Each of these subtrees contains attacks, both obvious and subtle, that have been suggested by various people over the last few years.

Ironically, the detailed attack trees for subverting electronic voting systems were not particularly important for this study. No voting system has adequately dealt with many of the attacks in this tree. We know these vulnerabilities exist in all systems and that this part of the attack can be performed by very few people, so we did not spend significant time evaluating computer security in Marin County. We are confident that elections cannot be made significantly more secure (in terms of ATS) with limited resources by focusing on computer security. Hence, we chose to look at other aspects of election security.

2.4 Input parameters

In addition to the attack tree, the reusable threat model has three groups of input parameters. The first group of parameters captures assumptions about a hypothetical election, specifically the number of votes cast and the margin of victory. The second group consists of two parameters that capture other assumptions in the model. The final group consists of about ten parameters that capture properties of the jurisdiction. The last

[6]Malware could be installed on individual machines in polling places, but this has a much larger attack team size and will always be dominated by other methods of malware insertion. There are so many vectors for malware insertion that we eventually decided not to try to list them comprehensively to avoid introducing unnecessarily complexity to the attack tree.

group of parameters is discussed in more detail in the next subsection.

The parameters about the hypothetical election (the first group) merit more explanation. We need to make assumptions about margin of victory, because it determines the number of votes that must be changed to change the election outcome. We want to focus on elections with margins of victory that are small, but not too small. We assume that changing the result of an election with a large margin would trigger suspicion, because the result would be so surprising, and that this would be a disincentive to an attacker with a goal of committing undetected fraud. On the other hand, if an election is too close, a tiny number of ballots is sufficient to change the outcome, and essentially any small-scale fraud can be successful (such elections are rare, although they often receive extensive media coverage). We believe it is most important to expend resources (including threat modeling) on the situations that are dangerous, and where trustworthiness can be significantly increased with reasonable effort. Those are the elections with margins of victory that occur fairly frequently in practice, where a fraudulent outcome would not raise too many questions, but where changes in election practices can make a significant difference in the difficulty of stealing the election without detecting.

To this end, we chose to assume two candidates and a 52%/47% margin of victory for modeling. This was the average margin of victory of contests of 2006 U.S. Senate races identified by the New York Times in 2006 as "Races to Watch" (Missouri, Montana, New Jersey, Pennsylvania, Rhode Island, Tennessee, and Virginia) [2]. The number of votes that must be stolen in order to change the outcome of a contest is a function of this assumed margin and the total number of ballots cast. Admittedly, choosing a specific margin of victory is suboptimal, and, in the future, it might be preferable to avoid assuming a specific margin by reporting ATS as a function of this variable. But we wanted to be able to report a single number for ATS. It is important to note that, although ATS may change with the margin, the ordering of attacks by ATS generally does not change. In other words, the greatest vulnerabilities remain invariant (especially as they are often attacks requiring an ATS of 1 for any small margin of victory).

There are two more parameters in the second group of assumptions: We assume that, to avoid an unacceptable risk of detection, the attacker will not attempt to steal more than a certain percentage of the votes in any individual precinct (15% in Marin) and a different percentage of the votes cast on any individual machine (20% in Marin). Bounds of these magnitudes are plausible, because the attacker would want to avoid an obvious skew in election statistics. These numbers are at the lower end of the range of possibilities (changes of less than this amount would almost certainly not be noticed, given the natural variation in precinct-by-precinct vote totals). The results of the next section are not particularly sensitive to these assumptions: Allowing larger percentage changes reduces the number of precinct results that have to be altered, and reduces the attack team sizes proportionally – but only for attacks that have relatively large attack teams under the current assumptions. So, the most dangerous attacks generated by AttackDog would continue to be the most dangerous attacks if these percentages were increased.

2.5 Applying the threat model to a local jurisdiction

We call the process of tailoring a reusable threat model to a particular jurisdiction "applying" the threat model. The basic tasks are looking for omissions in the threat model, and determining the value of input parameters for the threat model.

Looking for omissions in the threat model

An obvious worry about the reusable threat model is that possible attacks may have been overlooked. This may be less of an issue if election threat evaluations become routine, but, as of now, it is likely that new potential attacks will be discovered while studying the details of election operations in a particular jurisdiction.

While there is no recipe for finding all such attacks, there are ways to expose many of these issues. One of the most effective is to ask elections office staff where they think the vulnerabilities are ("What would be the best way to steal an election here?"). However, the primary method for exposing new attacks is to have an expert observe election processes and thinking creatively about how the election might be attacked.

In Marin County, we found that we only had to extend our existing model in one instance. Unlike counties we were familiar with, Marin had had "drop-off centers" as part of the process of transporting election materials back to the storage facility. Poll workers from individual precincts transported their materials to the drop-off point, where they were aggregated and then trucked back to the storage facility. This caused us to add a node to the attack tree where ballots are intercepted and replaced between the drop-off center and the storage area. The new node has a different ATS expression, because the attack team could be the individuals involved in transporting materials from the drop-off points to the storage facility. During this phase of

ballot transport, fewer workers have control over many more ballots, so, as we see in the next section, this has an impact on the attacks with small ATS. In the course of writing this paper, the model has been revised and reorganized to be more presentable, but there were no more extensions because of Marin County.

Input parameters

The essence of applying the threat model is determining the jurisdiction-specific input parameters. In some cases, this is very easy (e.g., estimating the number of precincts, votes, etc.). In Marin County, parameters were set for the total number of estimated votes (112,095, based on voter turnout in past recent elections), the number of polling places (208), the number of optical scan machines in the jurisdiction (210),[7] average number of poll workers in a polling place (4), the number of drop off locations (10), the number of workers transporting ballots from drop off centers to the central area (4),[8] the number of people counting ballots per team in the manual audit (4) and the percentage of ballots audited (1%).

Evaluating election procedures

The most difficult aspect of applying a threat model is evaluating security-critical election procedures in the county. The threat model does not spell out the details of these procedures. Instead, it summarizes the security of procedures with parameters that can take on a small number of qualitative values. The criteria are described in separately in textual form, and we assume that an expert makes a determination about which parameter setting is appropriate.

There are two such parameters in our threat model: One captures the stringency of tamper evidence technology used in for ballot boxes, etc. and the type of post-election manual audit. For example, the parameter "AuditType" can have values "BasicAudit" and "Type2Audit." The manual audits mandated by California are basic audits (small sample sizes, less than completely rigorously defined procedures, etc.) There is a lengthy description of what is required for a "Type2Audit" which entails choosing sample sizes based on a pre-defined acceptable level of risk and generally more rigorous procedures. Very few election jurisdictions in the U.S. meet that "Type2Audit" and many do not meet the "BasicAudit" standard (sometimes due to a complete lack of auditing requirements). The attack team size for defeating the audit depends on the value of AuditType. There is a parameter "TamperEvidenceLevel" that reflects the effectiveness of tamper evidence technology and procedures, with levels "ConventionalTE" and "EnhancedTE." ConventionalTE uses commercially available tamper-evident seals, adhesive paper labels signed by poll workers, etc., which can be defeated by skilled individuals. "EnhancedTE" is a rarely met standard where, either by manually auditing immediately at the close of the election with observers present, or keeping stored ballots in public view at all times. We chose "ConventionalTE" for Marin County.

The most time-consuming aspect of evaluating procedures is to determine precisely what the procedures are. For example, we were very concerned with physical security of ballots, which depends on the details of how voted ballots are transported and stored, warehouse security, etc. Understanding these procedures requires consulting the documentation and asking election administrators, but information from both of these sources is insufficiently detailed and sometimes inaccurate in practice. Talking to local election activists, former poll workers, and lower-level staff can fill in some of these gaps. However, the definitive information about procedures comes from expert observation.

In this study, much of our effort was expended on studying the physical security of paper ballots in the polling place and vote-by-mail and the detailed procedures for manual post-election auditing of ballots. We observed the election process in Marin County California over a period of 4 weeks around the November 2010 General Election and interviewed election staff during this time as needed. These observations included some limited pre-election preparation, 16 hours and 4 polling places on Election Day, the ballot materials drop-off stations and transportation to the Marin Civic Center, the materials acceptance and tabulation process that night, vote-by-mail signature verification and counting, provisional ballot adjudication and the 1% manual tally audit.

3 Findings

After applying the reusable threat model to Marin by setting the various parameters, and making small changes to address an election process difference that we had not modeled previously, we generated a list of attacks using AttackDog. AttackDog tends to generate many similar attacks that have identical attack team

[7] Each jurisdiction also has an AccuVote in each polling place for voters with disabilities who are unable to mark a paper ballot with a pen. However, a tiny fraction of total ballots are processed by these machines, so they do not make at attractive target to an attacker who wishes to affect the outcome of an election, and do not affect the analysis

[8] Drop-off parameters were added for Marin, as explained above.

size and vary in only a few steps, so we describe important *classes* of the generated attacks.

3.1 Subvert technology to change votes and tamper with ballots for audit attacks

In this class of attacks, the attacker uses malicious software or hardware to cause electronic vote-counting equipment to change electronic copies of ballots and vote totals. AttackDog generates a list of many similar attacks with small attack team sizes. As was discussed above, the "Attack code or hardware" subtree has very detailed steps on how to design, develop, install, and trigger effective malware, which accounts for most of the variations in the attacks. A simple example would be malware installed by someone with legitimate access to the machines. However, we feel that a strong case can be made that there are multiple practical ways to insert malware into voting systems that require an ATS of one *outsider*. Electronic voting has been studied extensively, and we feel confident that there is no basis for arguing for a much larger attack team size for these steps. The only serious question about the attack team size for this attack revolves around the post-election audit process that is in place to detect machine fraud or error. More specifically, as described in Section 2.3, the "Attack code or hardware" node is an AND node, and, since the steps of corrupting a system require only a very small attack team, increasing security depends on auditing. The attacker goal in this case is "Render routine statistical audit ineffective." The rest of the discussion of this class of attacks is devoted to manual audits.

Manual audits in Marin County

Before discussing attacks in Marin County, it is necessary to describe manual ballot auditing in more detail. Post-election manual auditing is a process where batches of ballots (usually, a "batch" is all of the ballots in a particular precinct) are chosen at random and hand-counted to check whether the reported totals from the machines match the actual contents of the paper-based systems. Manual audits rely on having a "voter-verified paper ballot" – a paper record of the vote that has been confirmed by the voter to have the correct votes, without depending on an electronic system. In Marin County, all voters use voter-verified ballots, which they fill out themselves. The ballots are then scanned electronically and counted. Voters can vote in polling places, in which case the voters deposit their ballots in a precinct-count optical scan system that counts ballots in the polling place. Voters can also vote by mail, in which case the ballot envelope is opened when delivered to the election office, and the ballots are counted on high-throughput scanners. After being counted, these ballots are stored in a ballot storage area controlled by the Registrar of Voters. They will be manually audited and possibly recounted later.

California election law requires manual auditing of (at least) 1% of the precincts in the county for each contest. Effective auditing procedures can be surprisingly subtle [9, 11, 12]. First, a "commitment" to precinct totals needs to be made. In Marin County, the precinct totals are printed and held by the audit supervisor. Then precincts are chosen randomly. In Marin County, dice are rolled to choose the precincts for each contest on the ballot. Then, boxes of ballots from those precincts are retrieved from the ballot storage area, and teams of four election workers count the ballots for the chosen contests in each precinct. When each precinct is complete, the counts are compared with the committed totals for the precinct, and, if there is a disagreement, the cause is investigated. The ballot counters should not be aware of the total they are trying to match.

In our threat model, there are three major avenues of attack against manual audits. The first is for the attacker to change the votes in a minority of the precincts, and hope that they are not audited. Since there are 208 precincts in Marin, only three need to be audited. If 20% of the precincts have changed votes, the probability that one of them will be audited is approximately $.8^3 \approx .5$. Also, most current audit laws do not adequately detect and correct incorrect outcomes, which would further weakens the effectiveness of audits in the cases where errors are actually detected [12].

If the attacker wants a lower risk of detection, there are two other types of attacks: change the ballots or their contents, somehow, to match fraudulent electronic totals, or subvert the audit process itself, so that it ballot counts or matching process fails to catch a discrepancy.

Tamper with ballots

Changing the paper ballots could be done in the ballot storage area where ballots are stored between the close of polls or while the ballots are being transported.

Changing ballots in the ballot storage area One possible attack on auditing is to change the paper ballots or their contents ("Replace paper ballots with fakes"). Since the paper ballots are stored for several days, physical security of the storage area is an important defense against this attack. This has been a

concern for a long time, so the reusable threat model addresses it.

In this case, the threat model gives an ATS of two, because of the way the steps are combined. In making up the model, we assigned an ATS of one outsider computer hacker to each of the computer-hacking steps, and one outsider with building access for the ballot-changing. The second individual is either someone with building access who is a county employee but not an elections office employee, or someone with lock-picking skills. The cost accounting takes the maximum over all steps of each type of outsider, giving an ATS of two for the entire attack. However, it is important to understand that ATS estimates are not expected to be precise. The important point, whether the true ATS is one or four, is that only a few attackers are required.

Is this attack feasible in Marin County? After examining security measures and practices, we believe so. In Marin County, paper ballots are kept in a storage area controlled by the elections office. The storage area has a computerized card key system that tracks the use of individual card keys and alarms. However, the procedures for controlling access to the storage area do not increase the attack team size significantly. It is in a multi-use building with other non-elections personnel. The card key system is run on the same computers that are used for other purposes, which are ordinary personal computers running Windows. Thus, there are several people with access to this system who could defeat controls on making new card keys, and who could delete or alter electronic card-key logs. There are also mechanical keys that can be used to open doors, bypassing the card key system and the card key access log.

Another step in this attack is to create counterfeit paper ballots to replace the real ballots. Perhaps the most practical method is to obtain blank ballots and fill them out. At first glance, this would seem to require a large attack team, but one person with access to the storage area can easily move that many boxes of ballots, and the process of forging ballots can be automated. Automatic signature machines or pen plotters (available used on eBay) could be used to reduce the team size for filling out ballots to one or two.[9]

Changing ballots during transportation This attack is to access the ballots as they are being transported from polling places to the storage area on election night. California election law requires that two poll workers from each precinct transport election materials, including ballot boxes, from the polling place to storage, although they can drive separate vehicles. Ballot boxes are sealed at the close of the election.

Ballot transportation necessitated some of the small number of Marin-specific changes in the threat model, because Marin, unlike counties we were familiar with, has poll workers deliver the ballots to one of ten intermediate "drop-off centers." Ballot boxes are collected in the centers and then trucked to the central storage facility. The changes consisted adding several parameters for the number of drop-off locations and the number of workers transporting ballots from the drop-off location to central storage, and changing the formulas to calculate ATS for the appropriate node.

According to AttackDog, this step has an ATS of four in Marin because each team transporting ballots from the collection point has a driver and a ride-along, often a poll worker, who deliver the ballots and other election materials to the office of the register of voters after the polls close. We estimate that two of the ten vans would be sufficient so that accounts for the need to have four attackers involved in a switch on the road. The estimate is based on observation of procedures in Marin County, and it is incorporated into the threat model via jurisdiction-specific parameters discussed in Section 2.

This change has an impact on the attack team size for subverting audits in this way. When ballot boxes are transported directly by poll workers, the two poll workers will typically have under 750 ballots in their possession (precincts are limited to no more than than 1,000 eligible voters by California law). But, with the drop-off centers, much larger numbers of ballot boxes are in the custody of a small number of people for about 40 minutes. During this time, with the participation of all workers for a drop-off center, forged ballots could be substituted for the real ballots while these workers have custody.

3.1.1 Subverting audit procedures

Another attack strategy is to subvert the audit procedures so that a mismatch between the paper ballots and electronic records is overlooked. Two potential weak points are the random selection of precincts and the commitment of the vote totals before the audit.

The "Select only honest precincts" attack works by, first, changing votes only in a fraction of precincts, and then arranging that only the non-fraudulent precincts are audited, by subverting the random selection of precincts. True random selection establishes a lower bound on the probability of auditing one of the fraudulent precincts. In Marin County, the random selection is done by rolling dice. But this is done with only two individuals present (no other observers chose to at-

[9] See: http://www.signaturemachine.com/, for example.

tend). Since the entire audit can be subverted by these individuals, the attack team size is two. To succeed in the presence of observers, the observers would have to agree not to report the departure from procedures that they witnessed, which would add them to the attack team (as informed participants).

Another potential attack is the process of comparing the results of individual counts with the committed totals. In Marin County, the people counting ballots do not know the expected totals (this is important to avoid biasing the ballot counting). When a contest has been counted, the counters take the results to an audit supervisor, who looks up the proper total and says whether it is correct or not. A malicious audit supervisor could approve non-matching counts, allowing fraud to be overlooked.

3.2 Discard vote-by-mail ballots before tabulation

There is another simple attack that, according to AttackDog, has an ATS of one. VbM ballots come in for several weeks before counting begins. Early in the process, ballots are sorted so that audits performed at the precinct level will be easier to perform. The sorting machine is on a floor below the area were the majority of the ballot-processing is done.

The attack would be to discard ballots before tabulation that, based on the return address, are likely to contain votes for a candidate not favored by the attackers. VbM ballots are accessible by a single person when they are left overnight. Insiders who discard ballots could also alter any tracking records associated with the ballots.

There may be a similar attack where a small number of postal workers discard ballots for non-preferred candidates. We have not studied this because we have not had a chance to learn U.S. Post Office procedures.

3.3 Attack vote by mail using stolen registrations

There is another attack against VbM that requires only a small attack team (our analysis gives an ATS of one). This is a relatively complex attack that requires the attacker to create new registrations for large numbers of citizens who do not vote, then vote in their place using vote-by-mail. We discuss the individual steps.

Acquire database of residents

For this attack, lists of registered and unregistered voters are needed. In California, voter registration lists and voting records are available from individual counties, including Marin, to candidates, political parties, and scholars, for a modest fee.[10] These names can be filtered from larger lists of residents obtained from other sources to find individuals who can be registered. An insider with legitimate access to government records of personal data would have convenient access to this data. This attack would be especially convenient for (insider) employees of the California Department of Motor Vehicles, who have especially easy access to useful records, and the ability to register voters (pursuant to the National Voter Registration Act of 1993).

Register unregistered citizens

To register to vote in Marin, individuals are supposed to supply, in addition to name and address, either a DMV ID number or, failing that, the last four digits of their social security numbers, date and place of birth. This information is widely available in government and commercial organizations. Social security numbers are regularly stolen in bulk by electronic intruders.[11] Also, insiders in businesses, banks, credit reporting agencies, and the state and county governments have legitimate access to this information. Given this information, these individuals could be registered by filling out voter-registration forms *en masse*.

Vote by mail ballots are returned to the elections office in envelopes signed by the voter, and these signatures are compared with those on the voter registration application to authenticate the voter. An attacker could circumvent this defense by tricking the voter into supplying his or her actual signature, or the attacker can steal a digital image of the signature from the DMV or elsewhere, and using the voter's real signature on the voter-registration forms. However, an easier approach would be for the attacker to write a program to generate images that look like signatures and print them onto the voter-registration forms, since Marin County does not have an independent source of a voter's signature for voters who have never registered in the past.

Acquire ballots

It seems that the most challenging aspect of this class of attacks is acquiring blank ballots. In Marin, blank absentee ballots are reasonably tightly controlled. The envelopes for the ballots are assigned unique numbers, and that number associated with the intended recipient in a database. However, an appropriate insider in the elections office could both update the database and

[10] See http://www.co.marin.ca.us/depts/rv/main/Forms/PriceList.pdf

[11] See http://www.privacyrights.org/data-breach

abscond with the corresponding ballots, so that step requires an ATS of one in our analysis.

Attackers could have individuals or computer technology fill in the ballots as desired by the attackers, as with the attacks on auditing that involve counterfeit ballots. Attackers could easily distribute ballots to various public U.S. Post Office mailboxes at various locations on various days to avoid raising suspicions at the election office.

3.4 Alternative metric: Insiders are more costly

Objections are sometimes raised to ATS because it does not distinguish between insiders and outsiders, when intuition suggests that it would be more difficult to enlist the participation of insiders in election fraud. We chose to consider all attackers to have equivalent cost because it is the simplest option, given that we have no way to estimate relative costs.

However, AttackDog is very flexible about how it computes attack costs. To illustrate this, we evaluate the robustness of our results by changing the threat model to consider one insider to be of the same cost as *ten* outsiders. In this scheme, one outsider costs one "attack difficulty point", and one insider costs ten points.

The ATS for attacks on optical scan computers and audits with the revised metric remains one, but the attack involving the audit supervisor subverting the audit increases in ATS to ten, and the lowest cost attacks are those where the attacker relies on the low statistical power of the audit to avoid detection (with probability .5). Also, two outsiders can still subvert the audit by substituting counterfeit ballots.

Discarding VbM ballots becomes relatively less attractive with the new cost metric because it involves insiders.

Obtaining blank ballots in the VbM mass registration attack required one insider. Without an insider, we estimate that the same attack can be conducted laboriously by eight outsiders. In this attack, requests are generated for ballots to be mailed to many addresses. We assume that hotels, P.O. boxes, apartment buildings, college residences, etc. can be the destination of ten ballots per building, that an individual attacker can spend a hundred hours over several weeks collecting these ballots, and that it takes an attacker-hour for every ten ballots. Obviously, these assumptions are subject to debate, but we feel that it is a reasonable ballpark estimate.

4 Related Work

The work reported here is novel in several ways. First, the attack trees and the AttackDog tool that supports them have several features that promote reusability, and the quantitative evaluation of attacks is much more flexible than other attack tree methods (and that flexibility is used to avoid over-counting of attackers when a single person can perform multiple steps in an attack). Our methodology also has novel aspects. Although attack team size has been used informally (by one of the authors) in a previous study, it has not been used by others using structured methods to evaluate election security. The definition of reusable threat models has not been discussed, to our knowledge, and the focus on evaluation of security practices in a local jurisdiction after detailed observation and validation is also new.

Over the past decade, there have been a number of extensive evaluations of various types of voting machines [1, 13, 3, 14, 4], and the use of post-election audit procedures to audit voting machine behavior and use [11, 12, 6]. These studies have been extremely important for establishing the insecurity of computer systems used in elections, and they are the starting point for our work, which is simply assumes an ATS of one for computer systems and focuses on other safeguards. Responding to previous computer security studies, Halderman et al. [10] examined the state of election system security after the California Top-To-Bottom Review and concluded that in many cases the insecurity of voting systems placed too much pressure on the proper operation of election procedures. This seems primarily to be a response to defenders of DRE and internet voting systems, who tend to invoke procedural safeguards as an antidote for insecure computer systems. There is little overlap with the procedural issues we discuss.

An extensive body of research has been developed around process modeling, including both fault trees and threat trees.[12]

Attack trees were first proposed over a decade ago [18]. Barr et al. [7] proposed their use in U.S. voting system standards as a tool for manufacturers to better instantiate and express threat models for their systems. Part of their proposal includes a suggestion that attack trees be used for threat modeling. This document is a proposal for a methodology, so it does not develop a detailed attack tree. However, an attack tree for certification would tend to focus more on equip-

[12]Note that in all of this work, researchers distinguish "fault trees", "attack trees" and "threat trees". These terms reflect the researchers' desires to emphasize failures due to error and failures due to malicious activity or both.

ment specifications, while an attack tree for jurisdictions focuses largely on procedures. We agree with these authors that attack trees might be a more effective approach to specifying security requirements and certifying that particular products meet those specifications. Moreover, the reusability features of AttackDog would probably be valuable in such an application as well.

Raunak et al. [17] applied process modeling techniques using the Little-JIL process modeling language and the FLAVERS finite-state verification tool to reason about what conditions in the election process could result in incorrect procedural outcomes and how to protect against them. Continuing this work, Simidchieva et al. [20, 19] investigated formal analysis of fault trees produced by Little-JIL in order to aid continuous process improvement, developing formal methods to find single points of failure and mitigate them. This work is primarily focused on election problems that do not result from creative malicious attacks. Weldemariam and Villafiorita [21] have developed a body of work that involves formally modeling election processes and artifacts as "asset flows" in the input language for a model-checker, NuSMV. The Marin County project has brought home to us the importance of detailed modeling and auditing of procedures. Currently, evaluation of procedural security is quite difficult and somewhat subjective. Integrating formal process modeling into our proposed attack tree methodology seems very attractive. However, given our experience with developing and applying our threat model, we do not believe that it will be possible to define such models with sufficient precision to admit a fully formal approach to the larger threat modeling problem any time soon.

Several research groups have used probabilistic modeling in election security. Buldas et al. [8] used attack trees and game-theoretic reasoning to compare the relative security between the Estonian Internet Voting System and the U.S. Department of Defense's SERVE system, finding that a number of features of SERVE make it less secure than the Estonian system. These authors recognize the limitations of their approach as it requires assigning somewhat arbitrary probabilities throughout an extensive attack tree. We believe that there is no reliable way to estimate probabilities. The study did not have a way of comparing attack difficulty (just probability of occurrence). This study also assumed the non-existence of insider attacks. A number of the authors assisted in a study conducted by The University of South Alabama lead by Yasinsac, funded by the U.S. Election Assistance Commission, to model election procedures in detail to produce extensive threat trees, threat matrices and a Threat Instance Risk Analyzer (TIRA) tool, a Microsoft Excel-based spreadsheet system for working with these artifacts [5, 22]. Much of the input threat modeling and threat trees in this effort were provided by us, and correspondingly the EAC project worked to enhance and further vet these materials. Pardue et al. [16] extended the EAC work to develop quantitative calculations of overall risk using a parametrized risk equation that used Monte Carlo-driven perturbation analysis to estimate relative risk values. This work was very helpful to us in revising our threat model. However, the tool lacks the advantages of AttackDog. For example, their model has large amounts of repetition because of TIRA's inability to deal with parameterized subtrees. Furthermore, we do not believe that probabilistic modeling is appropriate in this context.

5 Discussion

We reported here an attempt to understand how to do systematic quantitative threat evaluations of local election jurisdictions. The results were encouraging, but this is only the initial feasibility study. More studies are needed, and the threat model and methodology need further improvement.

As the introduction said, we feel that a major barrier is the time and expense needed to do such evaluations. We have proposed *reusable threat models* as a potential way around that problem, and take the initial steps to develop a methodology for applying the model.

Our long-term vision is that the threat model will be extended and generalized, as a series of jurisdictions is evaluated, until very few changes are required with each additional jurisdiction. One of the most important questions we needed to answer in this study was whether large changes were required in the threat model. A threat model is only truly reusable if the changes to apply it to a new jurisdiction are not very extensive. Results were encouraging. In this study, the changes were quite small. We found that we had to add a node and a few formulas to deal with "drop-off centers", but that is all. We found that vulnerabilities were in the places we most expected. Although it might have been more exciting to discover surprising vulnerabilities, we were happy not to be surprised, because we hope that threat evaluation will become routine.

Even with reusable threat models, the difficulty of evaluating detailed election procedures remains. This problem will have to be addressed in other ways. Standardization and careful documentation of procedures across many election jurisdictions would make evaluation much easier. In other words, we need reusable election procedures. Perhaps integrating formal process modeling into our methodology [17, 20, 19]

would be helpful. Research and experimentation with more effective and simplified systems for physical security and manual auditing could be very valuable for simplifying threat evaluation, also.

Threat evaluation has inherent limitations. Since attackers can be arbitrarily creative, it is not possible to anticipate all attacks. However, we feel that, in the long run, our approach is more likely to help expose vulnerabilities than most others, because a reusable threat model can accumulate years of experience and imaginative thinking by many people. Furthermore, the structured approach may help to expose gaps in reasoning.

There are limitations to the approach studied here. There are many potential problems with elections, only some of which we have addressed. Many problems are not under the control of election administrators, such as the conduct of campaigns. Many problems are not security problems. Machine failures or errors by poll workers create difficulties for voters, for example. Even within domain of security, we have limited the scope of problems we consider.

Although we have addressed the basic question of feasibility, there are major non-technical questions that need to be answered before we can realize the vision of routine election-system threat evaluations: "How can we cause evaluations to be done?" and "Who will do the evaluations?"

The first question is not easy to answer. We were fortunate to find a cooperative and supportive local election official who was interested in the results of the study. But, purely from the perspective of incentives, it is not clear why an election official would be motivated to invite someone to do an evaluation that may find problems that are potentially embarrassing and may require significant effort to fix. It may be necessary for someone, perhaps state officials, to mandate that such studies be done.

The answer to the second question is that the people conducting a study need independence and expertise. If the study is being sponsored by an entity that finds vulnerabilities embarrassing, threat evaluators may be selected who are less likely to find embarrassing vulnerabilities. If these studies are performed by consultants, the successful consultants may be those who are less than totally effective at finding vulnerabilities. Expertise by the evaluators is perhaps more important than the reusable threat model, although the two are closely related. Expertise is required to understand the threat model, but, more critically, to know how to apply it to a jurisdiction. For example, knowing what details of procedures are most important, and which need to be observed, is something that will require significant experience. How many expert evaluations can there be? Competition is usually desirable for many reasons (including cost effectiveness), but that would discourage sharing of threat models and expertise, which is fundamental to our approach. Perhaps this is a function that should be performed by a government agency. In any case, we would urge full public disclosure of models and the results of evaluations to promote sharing and to provide quality control through public accountability.

Finally, the primary goal of identifying vulnerabilities is to enable those vulnerabilities to be removed. Some of the vulnerabilities we have identified are easily addressed, while others are more difficult. For example, auditing of individual ballots instead of precincts could require much less time and labor for far superior detection of fraud or errors, which would also reduce vulnerabilities by making the process more robust. As another example, the problems of maintaining the integrity of paper ballots after they leave the voters' custody, especially in vote-by-mail, needs to be considered in much greater depth than it has been to date.

Acknowledgments

This work would not be possible without the election staff and volunteers of Marin County, especially Registrar Elaine Ginnold, Melvin Briones, Anthony Aquilino and Diane Belben.

We are grateful to Matt Bishop, Sean Peisert and anonymous referees for discussion and comments on earlier drafts.

We extend special thanks to Jim March, who reviewed early versions of the attack tree; Tim King, who contributed to the early design of AttackDog and tree development; Ion Sancho, Supervisor of Elections in Leon County, Florida, who contributed an elections official perspective to early versions of the tree; Philip Stark, for helpful discussions of threat models; and Sonal Mittal, who helped research background on threat trees and methodology.

This work was a collaborative effort of members of "A Center for Correct, Usable, Reliable, Auditable and Transparent Elections (ACCURATE)," funded by the National Science Foundation under Grant Numbers CNS-0524155, CNS-0524111, CNS-0524745.

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

Supplementary information

Details of the threat model are available at `http://verify.stanford.edu/MarinThreatModel.html`.

References

[1] Trusted agent report: Diebold AccuVote-TS voting system. Tech. rep., RABA Technologies, 2004.

[2] Races to watch: The senate. *The New York Times* (Nov. 2006).

[3] Top-to-bottom review of california's voting systems. California Secretary of State, Aug. 2007.

[4] Software reviews and security analyses of florida voting systems. Florida State University's Security and Assurance in Information Technology Laboratory, Feb. 2008.

[5] Election operations assessment: Threat trees and matrices and threat instance risk analyzer (TIRA) (DRAFT). University of South Alabama for the U.S. Election Assistance Commission, Dec. 2010.

[6] ANTONYAN, T., NICOLAOU, N., SHVARTSMAN, A. A., AND SMITH, T. Determining the causes of AccuVote optical scan voting terminal memory card failures. In *2010 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections, Washington, DC* (2010).

[7] BARR, E., BISHOP, M., DEFIGUEIREDO, D., GONDREE, M., AND WHEELER, P. Toward clarifying election systems standards. Tech. Rep. CSE-2005-21, Department of Computer Science, University of California at Davis, 2005.

[8] BULDAS, A., AND MÄGI, T. Practical security analysis of E-Voting systems. In *Advances in Information and Computer Security*, A. Miyaji, H. Kikuchi, and K. Rannenberg, Eds., vol. 4752 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2007, pp. 320–335.

[9] CORDERO, A., WAGNER, D., AND DILL, D. The role of dice in election audits—extended abstract. *IAVoSS Workshop on Trustworthy Elections 2006 (WOTE 2006)* (June 2006).

[10] HALDERMAN, J. A., RESCORLA, E., SHACHAM, H., AND WAGNER, D. You go to elections with the voting system you have: Stop-Gap mitigations for deployed voting systems. *USENIX/ACCURATE Electronic Voting Technology Workshop* (Aug. 2008).

[11] HALL, J. L. Improving the security, transparency and efficiency of california's 1% manual tally procedures. *USENIX/ACCURATE Electronic Voting Technology Workshop 2008* (July 2008).

[12] HALL, J. L., MIRATRIX, L. W., STARK, P. B., BRIONES, M., GINNOLD, E., OAKLEY, F., PEADEN, M., PELLERIN, G., STANIONIS, T., AND WEBBER, T. Implementing Risk-Limiting Post-Election audits in california. *Electronic Voting Technology Workshop/Workshop on Trustworthy Elections 2009 (EVT/WOTE 2009)* (Aug. 2009).

[13] KOHNO, T., STUBBLEFIELD, A., RUBIN, A. D., AND WALLACH, D. S. Analysis of an electronic voting system. *Proceedings of the IEEE Symposium on Security and Privacy* (2004), 27–40.

[14] MCDANIEL, P., BLAZE, M., VIGNA, G., BUTLER, K., ENCK, W., HURSTI, H., MCLAUGHLIN, S., TRAYNOR, P., AVIV, A., ČERNÝ, P., CLARK, S., CRONIN, E., SHAH, G., SHERR, M., KEMMERER, R., BALZAROTTI, D., BANKS, G., COVA, M., FELMETSGER, V., ROBERTSON, W., VALEUR, F., HALL, J. L., AND QUILTER, L. Everest: Evaluation and validation of election-related equipment, standards and testing (academic final report), Dec. 2007.

[15] NORDEN, L., AND LAZARUS, E. The machinery of democracy: Protecting elections in an electronic world: Brennan center task force on voting system security. Brennan Center for Justice at NYU School of Law, 2006.

[16] PARDUE, H., LANDRY, J., AND YASINSAC, A. A risk assessment model for voting systems using threat trees and monte carlo simulation. In *Requirements Engineering for e-Voting Systems (RE-VOTE), 2009 First International Workshop on* (2009), p. 55–60.

[17] RAUNAK, M., CHEN, B., ELSSAMADISY, A., CLARKE, L., AND OSTERWEIL, L. Definition and analysis of election processes. *Software Process Change* (2006), 178Ð185.

[18] SALTER, C., SAYDJARI, O., SCHNEIER, B., AND WALLNER, J. Toward a secure system engineering methodolgy. In *Proceedings of the 1998 workshop on New security paradigms* (1998), ACM, pp. 2–10.

[19] SIMIDCHIEVA, B. I., ENGLE, S. J., CLIFFORD, M., JONES, A. C., ALLEN, B., PEISERT, S., BISHOP, M., CLARKE, L. A., AND OSTERWEIL, L. J. Modeling and analyzing faults to improve election process robustness. In *2010 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections, Washington, DC* (2010).

[20] SIMIDCHIEVA, B. I., MARZILLI, M. S., CLARKE, L. A., AND OSTERWEIL, L. J. Specifying and verifying requirements for election processes. In *Proceedings of the 2008 international conference on Digital government research* (2008), pp. 63–72.

[21] WELDEMARIAM, K., AND VILLAFIORITA, A. Procedural security analysis: A methodological approach. *Journal of Systems and Software* (2011).

[22] YASINSAC, A., AND PARDUE, H. A process for assessing voting system risk using threat trees. In *Conference on Information Systems Applied Research 2010* (2010).