Vote Early, Vote Often, and VoteHere: A Security Analysis of VoteHere

Philip E. Varner(*pev5b@virginia.edu*)

April 26, 2001

Abstract

Recently there has been much academic and popular discourse about sociological and political ramifications conducting public elections over the Internet. According to the many groups and individuals wishing to profit from online voting, the technical problems are solved, and only the political and sociological ones need debate. However, this is far from reality. There currently exist significant, insurmountable technical hurdles which make an adequately secure and private Internet voting system impossible. This thesis demonstrates the many potential security and privacy problems with one of the most widely used systems, VoteHere from VoteHere.net.

I first present background information on voting, cryptographic voting protocols, and available system implementations. I then present FaSSAMM, the Fairly Simple Security Analysis and Modeling Methodology, a technique for easily conducting security analyses. This process is then used to analyze the documented VoteHere system for vulnerabilities. The result is an easy to understand demonstration of security problems with Internet voting which can be easily extended or detailed further. After conducting this security analysis, I believe even more adamantly that Internet voting should not be used for governmental elections now or in the near future.

0.1 Preface

This thesis has been a long and arduous process. I began work on it in the Summer of 2000 with the intention of applying voting system research to the area of online performance evaluations. I soon realized that this was not an adequately interesting project, and decided to focus on ballot distribution implementation in online voting systems. After a significant amount of research, I came to believe strongly against Internet voting for public elections. After several more topic shifts, I decided that the most useful course of action would be to show why Internet systems are not secure and should not be used. This is the result.

I would like to thank the following people: David Evans, my technical advisor; Edmund Russell and Mark Shields, my TCC advisers; VoteHere, for actually publishing system documentation; and finally, my parents, for their continuing support of my educational endeavors.

Contents

Chapter 1

Introduction

> "Remote Internet voting systems pose significant risk to the integrity of the voting process and should not be fielded for use in public elections until substantial technical and social science issues are addressed."
>
> – Internet Policy Institute report, [IPI01]

Recently there has been much academic and popular discourse about sociological and political ramifications conducting public elections over the Internet. According to the many groups and individuals wishing to profit from online voting, the technical problems are solved, and only the political and sociological ones need debate. However, this is far from reality. There currently exist significant, insurmountable technical hurdles which make an adequately secure and private Internet voting system impossible. Although the Internet may allow online voting, also known as ivoting, governmental elections are too important to be conducted over it. In this thesis, I demonstrate the many security and privacy problems with one of the most popular

systems, VoteHere from VoteHere.net.

This document presents an external security analysis of the VoteHere online voting system. This includes an attack tree analysis, attacker models, abuse cases, and detailed descriptions of the most likely and damaging attacks. This paper describes the many potential security problems with VoteHere and online voting systems in general, and why ivoting should not be used for governmental elections. This provides a basis for further security analysis, a standard against which to judge other systems, and a technical reasoning against Internet voting. The VoteHere system is not particular among Internet voting systems for inherent security and privacy problems. Most of these problems are present in all online voting systems, and VoteHere is simply used as a concrete example.

1.1 Voting

In modern democratic society, one of the most important duties of citizenship is voting. Many people, around half in national elections, find this too inconvenient and choose not to vote. Traditional voting is time consuming, expensive, and inconvenient and any solution to these problems seems as if it should be used. With the proliferation of the Internet, many believe this process can be improved upon significantly. There are many problems associated with Internet voting which are largely ignored or overlooked, accidentally or intentionally, in the name of looking good for governmental agencies or profit motive for voting service and equipment vendors.

Before 1890, secret ballot elections were rarely used. The ballot a voter

cast was usually public information, which caused to many problems. Voters can be coerced, either constructively or destructively, to vote a certain way. With the advent of the "Australian ballot", commonly called the secret ballot, this problem was significantly reduced. With secret ballots, a vote is receipt-free, meaning a voter gets no proof of how they voted, and anonymous, since there is no way to link a specific vote to a specific voter. This is more democratic and allowed for more valid elections.

In traditional systems, a voter usually goes to a designated polling station to receive a ballot cast his vote. After direct person-to-person verification, i.e. presenting driver's license to a voting authority, the voter is permitted a single, anonymous, irrevocable, receipt-free vote. When this is done, the voter is then given a digital ballot which allows a single vote. Once the ballot is used, it cannot be used again. However, this ballot must also be anonymous. The ballot must identify the voter as being permitted to vote, but not reveal their actual identity, and the voter must also be given assurances of this.

Traditional polling methods rely on a number of trusted parties to conspire to change the output of an election. This distributed trust system between election officials, political party officials, and poll watchers significantly deters cheating and corruption and helps ensure fairness of an election. Current methods require an attacker interact directly with the voting process to disrupt it, which means physical evidence and greater chance of getting caught, with substantial penalties as deterrents. However, the Internet is much more difficult to control, and therefore much harder to secure. Network-based attacks can be impossible to trace and even harder to inves-

tigate and prosecute. Disruption of the voting process is no longer limited to Democrats or Republicans, but is accessible to everyone from a 15-year old "script kiddie" to hostile foreign governments. An attacker no longer has to touch a ballot or go to a poll station. Attacks can be automated and run throughout the election period from around the world, which is a much more significant problem.

If an online voting system is going to be widely deployed, it must be easy to use. A large portion of the voting public has little or no knowledge of computers. Because of this, such systems must be much easier to use than most modern cryptographic software for these non-technical users. In many implementations, security is usually sacrificed to make the system easier to use. In a networked environment, a Web browser is a prevalent and known piece of software which is often utilized to make a voting system easier to use. However, security is often sacrificed to ease of use.

In conjunction with physical polling stations, many elections also offer absentee voting. This allows those who are not able to go to their polling station on the election day to still vote. This increases the mobility of the process, but in turn reduces privacy guarantees and increases the chance of fraudulent voting. In most states, absentee voting only makes up a small portion of the votes and therefore any corruption effort using it would either not affect the election (except in very close elections, such as in Florida during the 2000 election) or quickly be discovered. There is still a physical voting record artifact, the ballot, which can be recounted in case of problems and people cannot practically duplicate ballots. These protections give much better assurances of validity than a digital ballot that can easily be copied,

corrupted, or deleted in the process.

In a recent poll conducted by the Public Policy Institute of California, over half of respondents 18-44 years of age favored Internet voting. Of these respondent, only around 30% responded that they "never" have access to the Internet and/or email. Unfortunately, this has done much to encourage ivoting initiatives. Many companies wishing to profit from Internet voting which have been pushing somewhat recklessly for implementation on to narrow a time frame. The issues behind ivoting need to be examined conservatively before such potentially dangerous moves are made. In a voting system, privacy and security are desired, but are not always simultaneously achievable at a reasonable cost. In online voting systems, verification is very difficult to do accurately, and anonymity is difficult to ensure. This document shows some of the many problems with practical ivoting and why public elections are too important to trust to it.

Chapter 2

Literature Review

2.1 Public Key Cryptography

The idea of public key cryptography was first introduced by Diffie and Hellman in [DH76]. In 1978 and 1979, Rivest, Adelman, and Shamir published their seminal papers on public key cryptography which explained a practical method for implementation based on the difficulty in factoring large prime numbers [RSA78, RSA79]. The system uses a public key and a private key, instead of single key. The public key is available to anyone, who can then use it to encrypt messages that can only be decrypted by the private key. Conversely, the private key holder may digitally "sign" a message with the private key, which can then be verified with the public key. Digital signatures are usually used for authentication in online voting systems. For more information, see [RSA78, RSA79, DH76, St99, Sc94].

2.2 Homomorphic Encryption

Homomorphic encryption is a special type of cryptography in which the sum of two encrypted values is equal to the encrypted sum of the values. The high level formula for this is $E(a)+E(b) = E(a+b)$. In simple mathematics, this is equivalent to the communicative property of multiplication, where $a * b + a * c = a * (b + c)$. For a majority of cryptographic algorithms, this does not hold true. In most cases, it is undesirable because it may help reveal information which can be used to break the encryption. However, this is a desirable property if one wishes to have the sum of a group of encrypted values verified without revealing those encrypted values. In voting protocols, this is used to verify the tally of the ballots without revealing what those ballots are. More information about homomorphic encryption and its application to voting can be found in [Ne00c, Ad00d].

2.3 Zero Knowledge Proofs

Zero knowledge proofs are a method of proving a piece of information is known without revealing what that information is. Suppose Alice knows some piece of information X. Alice wants to prove to Bob that she knows X, but doesn't want to tell Bob X directly. So, Alice and Bob agree on a function $H()$, and Alice computes $Y = H(X)$ and tells Bob Y. This is then done multiple times with different $H()$ functions to reduce the chance of Alice getting Y correct without knowing X. The most well-known simple examples of this are the cave scenario and the card game scenario. Additional

information may be found in [Sh79, Sc94].

2.4 Cryptographic Voting Protocols

Many of the practical aspects of online voting are contained in David Chaum's paper "Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms" [Ch81]. However, Chaum was primarily interested in secure communications and makes no mention of voting. Since this paper, there has been a tremendous body of research specifically focusing on applying cryptographic principles to voting. There are three main schemes for cryptographic voting protocols: self-adjudicating, central voting authorities, and multiple voting authorities.

Self-adjudicating protocols are usually based on multiparty computation. In general, they do not scale well and are therefore not practical for elections involving more than a few voters. Examples may be found in [GMW87, BGW88, CCD88]. Therefore, the majority of implementable voting protocols take the security risk of using voting authorities. This is an acceptable risk, since most groups wishing to conduct unfair elections are usually not covert about it or give other indication of corruption. Using a single, central voting authority also may present problems. It can submit its own votes for abstainers, or drop those ballots it doesn't like. To distribute power and risk, most protocols use multiple voting authorities. A voting system is divided into a Validator and a Counter. The Validator authenticates a voter's right to vote and then performs some operation, like signing a ballot, to allow the voter to vote. The voter then sends this anonymous vote to the

Counter, who then checks the authenticity of the ballot and counts the vote. The premise behind this is to separate authentication and voting to preserve the voter's privacy. This however, does require some level of proof that the two agencies do not collude to breech a voter's privacy.

There are two main approaches to communication with the authorities: sending the completed ballot to the authorities encrypted and using anonymous communication channels. Schemes using the first technique can be found in [CF85, BY86, Iv91]. According to [FOO92], "these schemes are less practical for large scale elections, since it takes a lot of communications and computation overhead when the number of voters is large." Using anonymous communication channels, Chaum and Ohta proposed independent schemes in 1988 [Ch88, Oh88]. In both, the ballot tallying authority can immediately open ballots upon receiving them, and can therefore leak intermediate results. Also, a voter must reveal her vote to prove that it was not counted correctly which violates privacy concerns. Schemes to correct these problems can be found in [AMI91] and [Sa92], but also introduce other problems.

Most recent cryptographic voting protocols are based on the protocol presented in "A practical secret voting scheme for large scale elections" by Fujioka, Okamoto, and Ohta [FOO92]. They define requirements of a secure election system as: completeness, all votes are counted correctly; soundness, a dishonest voter cannot disrupt voting; privacy, all votes must be secret; unreusability, no voter can vote twice; eligibility, no one who isn't allowed to vote can vote; fairness, nothing must affect the voting; and verifiability, no one can falsify the result of voting.

Their protocol was the first practical method to theoretically fulfill these

requirements. A voter first fills in a ballot. They then use a blind signature technique to get the verification authority's signature. The voter then sends the signed ballot to the tabulation authority anonymously. The tabulation authority then publishes a list of the received ballots. At the end of the voting period, the voter opens his vote by sending his encryption key to the tabulation authority anonymously. The tabulation authority decrypts and counts the ballots, and announces the result.

The main modification most implementations (such as Sensus and EVOX) make is combining the Opening phase as part of the Voting phase. If the ballot collector and the key collector are separated, the voter can send the two things at the same time. In addition, voters can verify their vote was counted after the Counter publishes the ballots, but this is not required.

In 1994, Benaloh (formerly Cohen, name changed after marriage in 1986) and Tuinstra extend [FOO92] by making it receipt-free [BT94]. Jan and Tai present a method for voting with smartcards using a protocol similar to [FOO92] in [JT97], but is currently impractical. In two separate papers, Okamoto extended the work of Benaloh and Tuinstra in the area of receipt-freeness. His modifications make it more secure and practical for large systems [Ok96, Ok98], but introduce much complexity into the system. Karro and Wang improve security by correcting some problems with undetected vote replacing in their paper "Towards a practical, secure, and very large scale online election" [KW99]. A fairly complex but secure method was developed by Riera and Borrell in their paper "Practical Approach to Anonymity in Large Scale Electronic Voting Schemes" [RB99]. These works are all quite valuable, but add unnecessary complexity. Therefore, [FOO92]

with slight modifications is generally regarded as the best voting protocol.

2.5 Academic Implementations

There are many voting system implementations, but very few which adequately fulfill the requirements in [FOO92]. Ones which do not include Pericles from MIT and the Princeton USG system [DNW99]. Sensus [CC97] was developed by Lorrie Cranor and Ron Cytron at Washington University at St. Louis and is based on [FOO92]. This system fulfills the security requirements in [FOO92], but is not convenient for the voter. It also requires an in-place public key system, which is not practical for general public elections.

EVOX [He97] is one of the most promising projects. The main goals of the system are to be secure according to [FOO92], user-friendly so it is very easy to vote, stand alone so already in place structures are not required, and able to be used by on the order of a thousand voters per election.

2.6 Commercial Implementations

There are three main commercial implementations: SafeVote, election.com, and VoteHere. SafeVote and election.com have very little public information about the cryptographic protocols and technical implementation behind their system. Because of this, they should not even be considered to be adequate. SafeVote seems to believe that holding "hacking contests" against its system and releasing voter anecdotes is enough to assure people their system is secure. Because of this, VoteHere is the only system which is open enough

to be analyzed adequately.

2.7 Description of VoteHere System

2.7.1 System Design

VoteHere primarily utilizes the concepts of digital signatures, zero knowledge proofs, and homomorphic encryption for their cryptographic voting protocols.

A VoteHere election consists of three primary parts: election initialization, voting, and tabulation [Ne00b]. First, the Election Tabulation Authorities create a shared election key, of which each privately holds a portion. All Authorities must contribute their piece for vote decryption at the end of the process. The official ballot is then created and signed by each of the Authorities.

First, the voter electronically obtains this ballot, and validates its authenticity by checking the signature of the Authorities. The voter then uses software to create a voted ballot containing his choices. This voted ballot is then homomorphically encrypted using the Authorities' key. The voter digitally signs this encryption, and sends it to the vote collection center. The collection center then checks the digital signature against a database of valid signatures and performs a zero knowledge proof with the client. This proof ensures that the voter correctly completed the ballot, since the results cannot be seen until the end of the election. If it is valid, it writes the ballot to permanent media. The collection center then sends a receipt (containing no

information about the contents of the vote) back to the voter. If the voter does not receive a receipt, they may continue to attempt to vote until they do receive one.

At the end of the voting period, the votes are decrypted. All votes are first grouped together appropriately and used to compute the encrypted tally. Because of the homomorphic encryption properties, this tally can then be decrypted to reveal the actual tally. Each Authority then uses their piece of the key to decrypt the tally and revel the official election results.

2.7.2 Implementation Details

VoteHere may be setup as a remote, vote-from-home system or for use in a traditional polling station setup. In the Thurston County trial [Wy00], the election was run from the main VoteHere.net website as a remote setup. For large scale use, the system will most likely be run on multiple computers which are geographically distributed. The system is built using Microsoft Internet Information Server 4.0 on Windows NT 4.0. The system uses Active Server Pages for dynamic web page generation. The system works on Netscape Navigator 4.08 and above and Microsoft Internet Explorer 4.01 and above. The voter uses the web browser to go to a website indicated on their voter information card, which is mailed to them through the postal service. The voter is then asked for their voter identification string, which they type into an HTML form. The voter then fills out their ballot using a HTML form. The choices are then sent to a server which computes their ballot and sends it to the tabulation server. The voter then receives a receipt back from

the tabulation server on their web browser. Alternately, a browser plug-in can be downloaded to compute the ballot on the voter's local machine.

Chapter 3

Analysis Methodology

3.1 Abuse Case Modeling

3.1.1 Description

Abuse case modeling is a technique which adapts use case models to "capture and analyze security requirements in a simple way" [McDF99]. First, the actors on the system are enumerated. Examples include knowledgeable hackers, script kiddies, and foreign governments. The next step is to identify attacks on the system as abuse cases. These are then refined and expanded to model what the security mechanisms of the system need to defend against.

3.1.2 Problems

The main problem with abuse case modeling is that it is easy to miss abuse cases. There is no structured way to model the system and then base attacks on it. Abuse case models are done on an ad-hoc basis, without a cohesive

plan. This is an adequate approach for small systems, but make it easy to overlook elements in complex ones. Also, abuse cases are intended for use before system construction, and not as a security analysis tool. Also, there is little documented structure as to how an abuse case should be structured or detailed.

3.2 Attack Tree Modeling

3.2.1 Description

According to Bruce Schneier, "attack trees are a formal, methodical way of describing the security of systems, based on varying attacks" [Sc99]. This is done by first setting the attack goal, and then hierarchically decomposing system attacks. These decomposed elements are organized into a tree, with each non-leaf node being a subgoal and each leaf node being an attack. In addition, each attack is given a cost, in any number of categories. The defining factor of attack trees is the importance of associating risk with an attack. These include level of disruption, monetary cost, or special knowledge or equipment which is needed. If an attack costs more than the benefit, that attack will most likely not occur. However, if there are easy attacks which may result in benefit, then those need to be defended against.

3.2.2 Problems

The main problem with the attack tree methodology is that it does not present a total security modeling process, but rather a single tool. In [Sc99]

the reader is told simply to think things up and add them to the tree. This is adequate for small systems, but falls apart for complex ones.

3.3 Other Methods

Other methods for security analysis include threat trees and the Flaw Hypothesis Methodology [Mo00]. These are useful to varying degrees. The main problem with these methodologies is that they are not detailed enough and provide little direction for analysis. Threat trees are interesting, but are a superficial subset of attack trees. We therefore use threats as a basis for building attacks on the system. From the Flaw Hypothesis Methodology, we use the heavy reliance on documentation and specification. By looking at these artifacts, we can usually surmise direct security flaws or weak links that may be targets.

3.4 FaSSAMM: Fairly Simple Security Analysis and Modeling Methodology

3.4.1 Description

The main problem with most available security analysis techniques is that they are either highly technical and precise, or they are simply ad-hoc and non-comprehensive analysis tools. They are good for their purpose, so we therefore include them in a more complete approach. The following approach has been named the Fairly Simple Security Analysis and Modeling Method-

ology (FaSSAMM). FaSSAMM was developed by the author for conducting the security analysis in this paper. The main goal FaSSAMM is to enable the easy conduction of detailed security analyses in a structured way. This approach is still in early development, but worked quite well for this analysis.

The process begins with the available system documentation. When doing an external analysis, these are often unavailable. Few companies release detailed descriptions of their products, especially when dealing with security. We therefore hypothesize how a system is implemented from available documentation, system descriptions, and system interactions. This may lead to different attacks depending on different hypothetical designs.

We then define the system security requirements. The broad categories of security are confidentiality, integrity, and availability. These are then used to define concrete security requirements, like "votes cannot be altered" or "voters can only vote once." A detailed description of general security requirements can be found in [CoC98]. For an online voting system, we use the requirements in [FOO92] and add reliability.

The next step is to create an architectural overview diagram. This enumerates all systems and subsystems, interaction between systems, and actor interaction with systems. This should include all information gathered from available documentation, and hypothetical decisions about unknown information.

We then determine threats and where they can occur. We look at every element and connection in the system, and determine how we can attack that piece, either alone or in conjunction with another element. It is often easiest to look at these attacks through the eyes of an attacker, so we determine

the various broad categories of attackers as a single entity. An attacker description template can be found in Appendix C. These categories are usually delineated by skill level, resource level, and access level. We then take these threats and build abuse cases around them. An abuse case template can be found in Appendix E. These abuse case models are then used to build attack trees. This is an iterative process, so the construction of the attack tree may lead to new abuse cases, and vice versa. Combining these elements, we have a flexible, extensible security model which can be analyzed for deficiencies and solutions.

Chapter 4

Results

4.1 Discussion of Results

Using the methodology developed in the previous chapter, I performed an analysis on the documented VoteHere system. We follow [FOO92] and define a threats to the system as any action which affects the completeness, soundness, privacy, unreusability, eligibility, fairness, or verifiability of the voting. In addition, we add the practical element of reliability. Reliability means that all voting should not be stopped by any single point of failure. This is primarily concerned with system availability and the ability of attackers to disrupt the voting. In traditional voting, poll sites are autonomous and any single action cannot halt all voting. Reliability is a significant issue, since network congestion, network path outages, routing problems, power failures, and DNS problems can significantly hurt network availability and performance. This is one requirement which has largely been overlooked, yet is a major obstacle to ivoting.

System attacks mostly fall into two pairs of groups. Attackers may be internal or external to the system, and they may cause covert or overt damage to the system. Internal attackers are trusted individuals with special access to the system, such as administrators or tabulation authorities. External attackers are those who have no special authorized access to the system and may only access it through the same methods voters use. Attacks on the system are either covert or overt. Covert attacks occur and are undetected by the system or administrators. Overt attacks are attacks which it is evident that they occur but cannot easily be defended against. Covert attacks are obviously more dangerous, but overt attacks are also dangerous since voting cannot easily be reconducted.

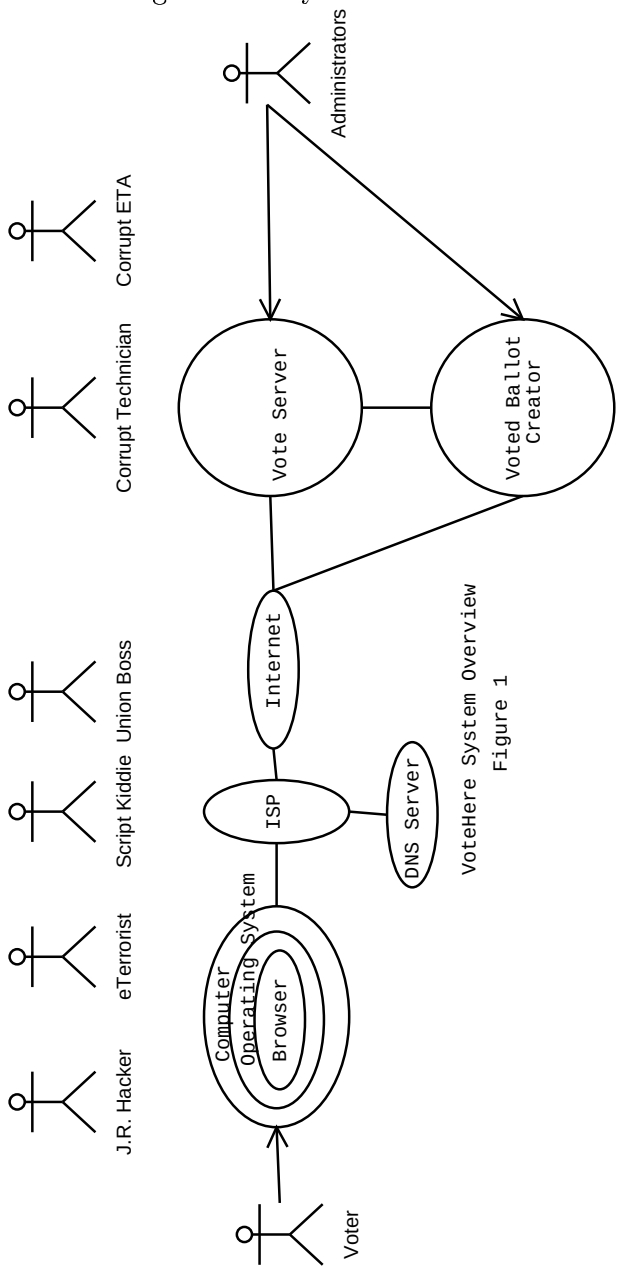
4.1.1 Discussion of System Model

The system model can be seen in Figure 4.1. The model shows all pieces of the system, and shows where the system actors interact with them. This model was created on a large drawing surface and the entire system interaction in [Ne00c] was enumerated on it.

4.1.2 Discussion of Attack Tree

The attack tree can be found in Appendix G. The eight categories of security violations formed the first level of the attack tree. I applied my knowledge of system attack to the determined attackers and system elements. I took all combinations of attackers and system security requirements and looked at the various pieces of the system to see how it could be attacked.

Figure 4.1: System Model.



After completing the attack tree, I found it was less important than I thought it would be. The abuse case models were much more useful, and the most damaging and dangerous attacks were obvious. The primary reason for the attack tree was to determine this, so it did not really fulfill its mission. However, it did help to enumerate all of the attacks which was useful.

4.1.3 Discussion of Attacker Models

The attacker descriptions can be found in Appendix D. The attacker descriptions were very useful in instrumenting system attacks. It was very useful to look at attacking the system though the eyes and resources of an attacker. These attackers represent the broad categories of attackers who would be interested in compromising a voting system. Most are also generally applicable to most other online systems. The one drawback to using an attacker-based approach is that some elements of the system may fail without being attacked, so these must also be enumerated.

4.1.4 Discussion of Abuse Case Models

The abuse case models can be found in Appendix F. The abuse case scenarios were very useful in developing attack descriptions. Because of their similarity to use cases, it was easy to transition my experience from that field. As I began elaborating on attacks with abuse cases, I found more categories of information which needed to be included and added them to the abuse case template. I found some of the abuse case model elements to be much less useful than others, but most were useful. The abuse case models are also

flexible, so they can be as simple or as detailed as the user needs. They also can take into account information which the user does not know and must hypothesize. Overall, the abuse case models where very useful and generally applicable to any security analysis.

4.2 Detailed Descriptions of Selected Attacks

4.2.1 Distributed Denial of Service Attack

This expounds upon Abuse Case A001. In any computer connected to the Internet, it is possible to overwhelm the system by sending it fake connection requests. The requests usually come from compromised machines, distributed throughout the Internet. The computer receives so many fake requests that it cannot process legitimate ones. This is known as a Distributed Denial of Service (DDoS) attack. There were several high-profile DDoS attacks against popular web sites such as cnn.com, ebay.com, and amazon.com in 2000. These attacks will be the most likely threat to ivoting, since a highly coordinated, well executed is impossible to defend against. Using a tool such as Trinoo or Tribal Flood Network, an attacker can configure compromised machines to be "zombies", and packet flood a target computer to disable it. If a voter is unable to connect to the vote server to vote, they may just not vote. This presents a significant problem to ivoting, since traditional polls always allow the voter to vote, even if they do have to wait in line for a small period of time. DDoS attacks will certainly occur against voting servers and present the most formidable challenge to ivoting.

4.2.2 Malicious Code Attack on Client

This expounds upon Abuse Case A003 and A004. One of the easiest attacks is to use malicious code to attack the voter's computer. This could come in the form of a Trojan horse, virus, or worm. A simple distribution method would be the in the form of an email worm, similar to the Melissa or LoveLetter macro viruses. This malicious code would attempt to email itself to any email address in the victim's address book. In an overt attack, a virus would release its payload on election day. The virus would disable or some how make the computer unusable for voting, therefore creating significant problems for the voter and for voting support technicians. A more covert attack would attempt to substitute its own ballot for the voter's. After a user executes it, it could run itself in the background on the voter's computer and wait for the voter to connect to the VoteHere web site. The virus would then substitute its own ballot when the voter submitted his. This may be found in the tabulation phase, since all votes would be publicly available, but would be difficult to correct. Malicious code is very difficult defend against. People will often do unsafe things, such as opening email attachments. These actions but secure systems in jeopardy and create large security violations.

4.2.3 Domain Name System Attack

One attack would involve attempting to corrupt the Domain Name System, or DNS. DNS servers are computers that translate addresses, such as www.votehere.net, into IP addresses, such as 206.253.221.100, for use in network communications. There has been a history of security problems with

BIND, the most popular DNS server program. The DNS system is arranged hierarchically so machines send queries for addresses they do not know further up the hierarchy to fulfill them. An attacker could break into the root DNS server for the votehere.net domain or the DNS server that handles requests for an ISP. They could then do several actions. They could crash the DNS server so that no one could resolve the needed address, or they could redirect requests for one domain to a fake address. By crashing the system, it would prevent all voters from voting for a time. This would be impossible with a traditional voting system, since polling sites are distributed and autonomous. If the attacker redirected voters, he could submit different ballots for them and corrupt the tally. This would be revealed at the end of the voting, but would require the voter to reveal their vote to correct it. Neither of these attacks could be widely successful in a traditional voting system, but present real threats to ivoting.

4.2.4 Attack by Corrupt Voting Administrator

There are many ways in which a corrupt voting administrator can compromise an election. They could submit fake ballots, intentionally shutdown equipment, erase ballots, or perform other detrimental activity. This can also occur in traditional voting systems, so it is not particular to ivoting. However, because traditional voting system have physical artifacts and physical actions associated with them, the distributed trust system between election participants is easier to maintain. When working with digital systems, it is easier to perform covert actions and subvert and election. Security is much

more difficult to ensure than traditional systems and should be considered a significant risk.

Chapter 5

Conclusion

The goal of this thesis was to demonstrate the numerous security and privacy vulnerabilities that exist in Internet voting systems. VoteHere was chosen as an example of this because they are the only company to provide enough system documentation to even consider analysis. FaSSAMM was developed specifically for this project, but as a generally applicable analysis methodology and worked quite well.

5.1 Interpretation

After this security analysis, I believe even more adamantly that Internet voting should not be used for governmental elections now or in the near future. This document shows some of the vulnerabilities which exist in online systems and cryptographic voting systems. The impetus to use the Internet for voting is large – ease of use, convenience, and cost savings. However, if the integrity of the voting system cannot be assured, than this reasoning is

fundamentally flawed.

The importance of voting is too great and risks too numerous for such a system to be instituted for large scale elections, such as for state and national government. It may be possible to use online voting for small-scale elections, such city or county government, but this issue must be examined further.

5.2 Recommendations

Further research needs to be completed on the security vulnerabilities of the VoteHere system and other online voting systems. Technical problems need to be revealed on paper before they are revealed during an actual election to avoid detrimental consequences of a failed election. Some people are advocating online voting in the 2004 Presidential election, and this needs to be vigorously rebuffed not only for political and sociological reasons, but also on technical grounds.

In the future, it may be possible to conduct large governmental elections online. However, this is highly unlikely. The requirements for voting are such that it is nearly impossible to fulfill them with anything other than a physical system. This thesis will hopefully provoke further debate on the technical problems with online voting systems and prevent their use in public elections.

Bibliography

[Ad00a] Adler, Jim. "Internet Voting Primer." http://www.votehere.net (Nov 2000).

[Ad00b] Adler, Jim. "Security versus Compatibility in Online Elections." http://www.votehere.net (Nov 2000).

[Ad00c] Adler, Jim. "Internet Voting Security." http://www.votehere.net (Nov 2000).

[Ad00d] Adler, Jim, et. al. "Computational Details of the VoteHere Homomorphic Election." http://www.votehere.net (Nov 2000).

[AMI91] Asano, T., T. Matsumoto, and H. Imai. "A Study on Some Schemes for Fair Electronic Secret Voting" (in Japanese). *The Proceedings of the 1991 Symposium on Cryptography and Information Security, SCIS91-12A.* (February 1991.)

[BGW88] Ben-Or, M., S. Goldwasser, and A. Wigderson. "Completeness Theorems for Non-Cryptographic Fault-Tolerant Distributed Computing." *Proceedings of the 20th Annual ACM Symposium on Theory of Computing.* (1988).

[BT94] Benaloh, Josh and Dwight Tuinstra. "Receipt-Free Secret-Ballot Elections (Extended Abstract)." *Proceedings of the Twenty-sixth Annual ACM Symposium on the Theory of Computing.* (May 1994).

[BY86] Benaloh, J., and M. Yung. "Distributing the Power of Government to Enhance the Privacy of Votes." *Proceedings of the 5th ACM Symposium on the Principles of Distributed Computing.* (August 1986).

[CC97] Cranor, Lorrie Faith and Ron K. Cytron. "Sensus: A Security-Conscious Electronic Polling System for the Internet." *Proceedings of the Hawai'i International Conference on System Sciences.*

[CoC98] Common Criteria Implementation Board. "Common Criteria for Information Technology Security Evaluation" (May 1998).

[CCD88] Chaum, D., C. Crepeau, and I. Damgard. "Multiparty Unconditionally Secure Protocols." *Proceedings of the 20th Annual ACM Symposium on the Theory of Computing.* (1988).

[Ch81] Chaum, D. "Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms." *Communications of the ACM.* (February 1981).

[Ch85a] Chaum, David. "Security Without Identification: Transaction Systems to Make Big Brother Obsolete." *Communications of the ACM.* (October 1985).

[Ch85b] Chaum, David. "Showing Credentials without Identification: Transferring Signatures between Unconditionally Unlinkable Pseudonyms." *Advances in Cryptology - AUSCRYPT '90 International Conference on Cryptology.* Springer-Verlag, Berlin. (1991).

[Ch88] Chaum, David. "Elections with Unconditionally-Secret Ballots and Disruption Equivalent to Breaking RSA." *Journal of Cryptology, Vol. 1, No.1.* (1988).

[CF85] Cohen, J. and M. Fisher. "A Robust and Verifiable Cryptograpically Secure Election Scheme." *26th Annual Symposium on Foundations of Computer Science, IEEE* (October 1985).

[DH76] Diffie, W., and M.E. Hellman. "New Directions in Cryptography." *IEEE Transactions on Information Theory, v. IT-22, n. 6.* (Novemeber 1976).

[DNW99] Davenport, Ben, Alan Newberger and Jason Woodard. "Creating a Secure Digital Voting Protocol for Campus Elections." http://www.princeton.edu/ usgvote/technical/paper.html (no longer available). (April 2000).

[Du99] DuRette, Brandon William. "Multiple Administrators for Electronic Voting." http://theory.lcs.mit.edu/ cis/theses/DuRette-bachelors.pdf (Nov 22 2000).

[FOO92] Fujioka, Atsushi, Tatsuaki Okamoto, and Kazui Ohta. "A practical secret voting scheme for large scale elections."

[GMW87] Goldreich, O., S. Micali, and A. Wigderson. "How to Play Any Mental Game or a Completeness Theorem for Protocols with and Honest Majority." *Proceedings of the 19th Annual ACM Symposium on Theory of Computing*. (1987).

[He97] Herschberg, Mark A. "Secure Electronic Voting Using the World Wide Web." Master's Thesis, Massachusetts Institute of Technology, June 1997. http://theory.lcs.mit.edu/ cis/theses/herschberg/

[Iv91] Iverson, K.R. "A Cryptographic Scheme for Computerized General Elections." *Advances in Cryptology - Crypto '91, Lecture Notes in Computer Science 576*. Springer-Verlag, Berlin. (1992).

[IPI01] Internet Policy Institute. "Report of the National Workshop on Internet Voting: Issues and Research Agenda." http://www.internetpolicy.org/research/e_voting_report.pdf (Mar. 2001).

[JT97] Jan, Jinn-Ke and Chih-Chang Tai. "A Secure Electronic Voting Protocol with IC Cards." *Journal of Systems & Software*. Elsevier, USA. (Nov. 1997).

[Jo00] Jones, Bill. "A Report on the Feasibility of Internet Voting." http://www.ss.ca.gov/executive/ivote/ (January 2000).

[KW99] Karro, Jared and Jie Wang. "Towards a practical, secure, and very large scale online election." *Proceedings 15th Annual Computer Security Applications Conference (ACSAC'99)*. IEEE Comput. Soc., Los Alamitos, CA, USA. (1999).

[McDF99] McDermott, John and Chris Fox. "Using Abuse Case Models for Security Requirements Analysis."

[Mo00] Moberg, Fredrik. "Security analysis of an information system using an attack tree-based methodology."

[Ne00a] Neff, C. Andrew. "Recounting an Electronic Election - What Does it Mean?" http://www.votehere.net (Nov 2000).

[Ne00b] Neff, C. Andrew. "Conducting a Universally Verifiable Electronic Election Using Homomorphic Encryption." http://www.votehere.net (Nov 2000).

[Ne00c] Neff, C. Andrew. "Example Homomorphic Election." http://www.votehere.net (Nov 2000).

[Oh88] Ohta, K. "An Electrical Voting Scheme using a Single Administrator" (in Japanese). *1988 Spring Convention Record, IEICE, A-294.* (March 1988).

[Ok96] Okamoto, Tatsuaki. "An Electronic Voting Scheme." *Proceedings of IFIP '96, Advanced IT Tools.* Chapman & Hall. (1996).

[Ok98] Okamoto, Tatsuaki. "Receipt-free Electronic Voting Schemes for Large Scale Elections." *Security Protocols. 5th International Workshop Proceedings.* Springer-Verlag, Berlin, Germany. (1998).

[RB99] Riera, Andreu and Joan Borrell. "Practical Approach to Anonymity in Large Scale Electronic Voting Schemes." (1999).

[RSA78] Rivest, R.L., A. Shamir, and L.M. Adleman. "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems." *Communications of the ACM*, v.21, m. 2. (Feb 1978).

[RSA79] Rivest, R.L., A. Shamir, and L.M. Adleman. "On Digital Signatures and Public Key Cryptosystems." MIT Laboratory for Computer Science, Technical Report, MIT/LCS/TR-212. (Jan 1979).

[Sa92] Sako, K. "Electronic Voting System with Objection to the Center" (in Japanese). *The Proceedings of the 1992 Symposium on Cryptography and Information Security, SCIS92-13C.* (April 1992).

[SMK01] Scambray, J., S. McClure, G. Kurtz. "Hacking exposed : network security secrets & solutions." Osborne/McGraw-Hill. (2001).

[Sc94] Schneier, Bruce. *Applied Cryptography.* John Wiley & Sons, New York. (1994).

[Sc99] Schneier, Bruce. "Attack Trees: Modeling security threats". Dr. Dobb's Journal December 1999. http://www.counterpane.com/attacktrees-ddj-ft.html (Feb 2000).

[Sh79] Shamir, Adi. "How to share a secret." Communications of the ACM, 22(11):612-613, 1979.

[St99] Stallings, William. *Cryptography and Network Security: Principles and Practice.* Prentice-Hall, Upper Saddle River, New Jersey. (1999).

[VH00a] VoteHere.net. "Key principles of election integrity." http://www.votehere.net (Nov 2000).

[VH00b] VoteHere.net. "VoteHere Election System Version 2.2 Platinum Vote User's Guide." http://www.votehere.net (Nov 2000).

[VIP00] Voting Integrity Project. "Voting Technology." http://www.voting-integrity.org/projects/technology/internetvoting (Nov 22 2000).

[Wy00] Thurston County Auditor: Internet Voting Trial Comprehensive Review. http://votehere.net/VH-Content-v2.0/ThurstonCountyAuditor/index.htm (Aug 2000).

Appendix A

Useful Resources

- SafeVote - www.safevote.com
- Voting Integrity Project - www.voting-integrity.org
- SecurePoll - www.securepoll.com
- Rebecca Mercuri - www.notablesoftware.com
- Internet Voting Technology Alliance - www.ivta.org
- Lorrie Cranor's Electronic Voting Hotlist - www.research.att.com/ lorrie/voting/hotlist.html
- Election.com - www.election.com
- VoteHere - www.votehere.com
- Internet Voting and Democracy Symposium at Loyola Law School - www.lls.edu/internetvoting.htm
- Internet Voting information at the Brookings Institute - www.brook.edu/gs/projects/iVoting.h

Appendix B

Glossary of Terms

Applet - a small program downloaded by web browser, usually written in the Java programming language

Authentication - the act of verifying the identity of an entity. In ivoting, this usually means electronically verifying via some piece of digital information that a person is who they say they are.

Browser - common term for a Web browser. A computer program used to navigate pages on the World Wide Web. Examples are Netscape Navigator, Microsoft Internet Explorer, NCSA Mosaic, Opera, and Lynx.

Ciphertext - text or other information which has been encrypted

Client - a computer or computer program which sends requests to a server computer or computer program.

Cryptography - the art and science of keeping messages secure

Decryption - using a key to decode an encrypted message

Digital Signature - a cryptographically verification proving a document was "signed" by a person or software with a piece of secret knowledge. This is usually used in the context of a person digitally signing a document to prove that it is from them.

Encryption - using a key to encode a message

ETA - Election Tabulation Authority. These are the people who admin-

ister the election and hold the pieces of the decryption key.

HTML - Hypertext Markup Language, a description language used for formatting web pages

HTTP - Hypertext Transfer Protocol, the communication protocol used for interaction between web browsers and web servers on the Internet

Internet - a worldwide network of computers. Note that this is merely the underlying physical architecture, not the software and communications protocols which run over it.

Internet voting - any method by which voters can use the Internet distributed computing network to cast ballots in elections.

ivoting - also spelled i-voting, stands for Internet voting.

IP Address - a unique numeric address assigned to every computer on the Internet, ex. 128.143.156.90

ISP - Internet Service Provider, a company who sells access to the Internet via phone line dial-up, Digital Subscriber Line (DSL), or cable modem

Key - a bit string used to encrypt and/or decrypt messages

Key Pair - in public key cryptosystems, this refers to the public key and corresponding private key

Malicious Code - also called malcode, a program with undesirable behavior. This can refer to intentionally malicious code, like a virus or Trojan horse, or to accidental software bugs which cause unintended problems

Online - a synonym for "on the Internet," usually refers to the World Wide Web

Packet - a small chunk of data with delivery directions transmitted over a network

PC - any commercially available computer marketed to individuals or businesses for use by a single person at a time

Plaintext - information in its original, unencrypted format. The word "text" is a misnomer, since plaintext can consist of any binary formatted data

Privacy - protecting personal data from being seen by unauthorized parties

Private Key - the key in a public key cryptosystem kept private by the using party. It is used for decrypting documents encrypted with the public key and by the owner for signing documents.

Protocol - a defined method for action. In the context of computer networks, it usually refers to algorithmic and data transfer procedures.

Public Key - the key in a public key cryptosystem released to the public. It is used to verify documents signed by the private key and to encrypt documents so only the private key owner can decrypt them.

Public Key Cryptography - a cryptographic method by which two keys are used, one which is made public and one which is kept private.

Security - general term covering issues of authentication, validation, integrity, etc.

Server - a computer which accepts requests from clients, processes them, and returns the results of such to the client.

Symmetric Key Cryptography - An algorithm which uses a single key to encrypt plaintext and decrypt ciphertext

Trojan horse - a program with has hidden malicious function. An example would be a program which says it will display dancing bears and does so, while secretly deleting the contents of your hard drive.

URL - Uniform Resource Locator, a alphanumeric address used to identify a computer on a network, ex. www.virginia.edu

Virus - a piece of malicious code which attaches itself to other programs and attempts to replicate itself. Although not required, viruses usually do

damage to a computer system or have other undesirable consequences.

Web page - a single document written in HTML, usually retrieved from a Web server over the World Wide Web by a web browser

Web spoofing - on the Web, to pretend to be a computer other than the one you are by using a similar, often typographically incorrect, host name

World Wide Web, Web, WWW - The system of web client and web server interactions using HTTP over the Internet.

Appendix C

Attacker Description Template

The basic template for a attacker description is:
Name: The descriptive name of the attacker.
Symbol: Short name by which they can be referred.
Resources: Resources available to the attacker, including computer.
equipment, network access, and other specialized equipment.
Skills: Level of computer security circumvention knowledge.
Access: Level of access to the system.
Risk: Level of risk the attacker is willing to take to accomplish their goals.
Comments: Additional information about the attackers

Appendix D

Attacker Descriptions

Name: J.R. Hacker
Symbol: JRH
Resources: Fast network connection, access to multiple hacked accounts
Skills: Extremely knowledgeable about network protocols, operating systems, security analysis, experienced hacker
Access: External Access only
Risk: Willing to take a moderate amount of risk, does not want to get caught
Comments: JRH is a knowledgeable attacker who can do a significant amount of damage.

Name: eTerrorist
Symbol: ET
Resources: Fast network connection, access to multiple hacked accounts, access
to physical security disruption methods (weapons, explosives, etc.)
Skills: Extremely knowledgeable about network protocols, operating systems, security analysis, experienced hacker (think Kevin Mitnick or Mudge)
Access: External Access only
Risk: Willing to take extreme levels of risk
Comments: ET is much more dangerous than JRH because of the level of acceptable risk. ET is willing to do nearly anything to disrupt voting, and does not care about costs. ET includes hostile foreign governments and radical domestic groups.

Name: Script Kiddie
Symbol: SK
Resources: Network connection, possible access to hacked accounts
Skills: Few. Mostly relies on pre-built "scripts" to attack systems, does not

have a significant amount of specialized knowledge.
Access: External Access only
Risk: Will take unnecessary risk, generally juveniles not worried about authorities, very numerous
Comments: SK is usually a younger attacker who is mainly interested in causing mischief.

Name: Corrupt Sysadmin
Symbol: CS Resources: No special ones.
Skills: Experienced with system and procedure
Access: Access to system internally
Risk: Willing to take risk, but not too much
Comments: Dangerous because of high trust levels and high system access

Name: Corrupt Election Tabulation Authority
Symbol: CETA Resources: Vote decryption key
Skills: Experienced with system and procedure
Access: Access to system internally
Risk: Willing to take risk, but not too much
Comments: Dangerous because of high trust levels and high system access

Name: Union Boss
Symbol: UB Resources: None
Skills: No technical
Access: Access to Voters
Risk: Protection system would prevent from getting caught, willing to take some risk
Comments: UB is primarily interested in getting a voter to vote for an endorsed candidate.
Name: Voter
Symbol: V Resources: None
Skills: None
Access: External Access only
Risk: None
Comments: Multiple voters attempting to access the system simultaneously may crash system. This is not malicious, but can cause problems for the system.

Appendix E

Abuse Case Model Template

ID: A symbolic identification number, such as A125
Title: A brief title for the attack.
Description: A one or two sentence description of the attack.
Harm: The harm level or levels incurred by a successful attack. This may assign different harm levels for various degrees of success. This can either be is a broad assignment such as "high" or "low" or a more precise measure of harm. of harm.
Attackers: Names of the attackers who can or would perpetuate this attack.
Visibility: Whether or not the attack would be apparent to observers. This is usually "high" in the case of an overt attack, such as a DDoS attack, or "low" for a covert, such as network traffic sniffing. Combinations of the two can be noted for varying levels of success.
Violations: Previously determined security requirements which this attack violates.
Likelihood: The likelihood that the attack would occur. This is very imprecise, and usually only serves to show that an attack is so obvious or easy that it would certainly occur.
Preconditions: Conditions which must occur to allow the attack to take place.
Triggers: Events by non-actors which cause the attack to occur.
Attack Flow of Events: An detailed description of the attack.
1. Event One
2. Event Two
3. Event Three
Alternative Paths: If there are slight variations on an attack which may produce different results, they may be shown here instead of in a separate abuse cases.
1a. Variation on step 1.
3a. Variation on step 3.

3b. Variation on step 3, number 2.
Postconditions: Conditions which are true after a successful attack.
Special conditions: An special conditions surrounding the attack.
Comments: An additional comments on the attack.
Defense Mechanisms: Ways to defend against the attack. These are brief descriptions rather than detailed defenses.
Additional Information: Resources (books, URLs) for more information about the attack or defense.

Appendix F

Abuse Cases

ID: A001
Title: Distributed Denial of Service (DDoS) attack against vote server
Description: Utilizing a distributed set of compromised "zombie" machines, an attacker floods the vote server with network connection requests, causing the vote server to be unable to service legitimate requests.
Harm: Potential to completely stop voting
Attackers: JRH, ET, SK
Visibility: High
Violations: Reliability
Likelihood: High
Preconditions: None
Triggers: None
Attack Flow of Events:
1. Attacker compromises computers attached to the Internet and gains full access.
2. Attacker uses a program such as Tribal Flood Network, Trinoo, or a custom program to "packet flood" the vote server.
3. Packet flood causes vote server to be unable to service legitimate requests.
Alternative Paths:
2a. Attacker attacks ISPs and prevent voters from being able to contact the vote server
Postconditions: Voting is halted.
Special conditions: None
Comments: This attack will definitely occur during the voting period.
Defense Mechanisms: Virtually impossible to defend against a well-planned, coordinated, global attack by JRH or ET. Attacks by SK can be successfully defeated, but may cause some brief slowdown or stoppage in the voting. Defense usually consists of packet filtering at the router level.

Additional Information:
CERT on DDoS, Trinoo, and TFN - http://www.cert.org/incident_notes/IN-99-07.html
Information on Trinoo - http://staff.washington.edu/dittrich/misc/trinoo.analysis
Information on TFN - http://staff.washington.edu/dittrich/misc/tfn.analysis

ID: A002
Title: Sudden spike in number of voters using system
Description: At some time during the voting, an overwhelming number of voters attempt to simultaneously use the system, thereby causing it to become unavailable or intolerably slow.
Harm: Potential to completely stop voting
Attackers: Voter
Visibility: High
Violations: Availability
Likelihood: Moderate
Preconditions: None
Triggers: None
Attack Flow of Events:
1. At some time, a large number of voters decide to use the system. The most likely times for this would be during lunch time, near the end of the work day, or near the end of the voting period.
2. Vote server is overwhelmed by number of requests.
3. Vote server either slows down to an intolerable level and voters give up, or vote server crashes and voting is halted.
Alternative Paths: None
Postconditions: Voting is halted or significantly slowed.
Special conditions: Many voters must attempt to vote simultaneously.
Comments: This attack would most likely not happen in a traditional voting system. If it were to, the voter would be able to determine the problem (i.e. a long line of voters waiting), and would wait to cast their vote.
Defense Mechanisms: Maximum cap on the number of simultaneous voters. This is not very good, since it makes the system less convenient and not always available.
Additional Information: None

ID: A003
Title: Malicious code attack - disable voter computer
Description: Malicious code attacks the voter's computer and causes it to be inoperable.

Harm: Significantly impedes voting
Attackers: JRH, ET, SK
Visibility: High
Violations: Privacy
Likelihood: High
Preconditions: Voter must be running vulnerable operating system and software.
Triggers: None
Attack Flow of Events:
1. An attacker creates a macro virus which propagates through email. This virus disables the computer by corrupting the operating system, the web browser, the networking setup, or some other piece of the computer vital to voting.
2. The attacker releases the virus into the wild.
3. Voters release the virus by opening an email attachment.
4. Voters cannot vote with their computer.
Alternative Paths:
1a. Other forms of malicious code could be used instead of macro viruses.
Postconditions: Voter cannot use their computer to vote.
Special conditions: Voter must be using Microsoft products susceptible to macro viruses.
Comments: This attack is guaranteed to occur
Defense Mechanisms: Educate users on the dangers of malicious code. Require voters to use anti-virus software. Require users to boot their computers from a "clean" boot disk.
Additional Information:
Gary McGraw and Greg Morrisett, Attacking Malicious Code: A Report to the Infosec Research Council, IEEE Software, September/October 2000.
http://dlib.computer.org/so/books/so2000/pdf/s5033.pdf
Information on W32/Naked@MM "Naked Wife" virus:
http://vil.nai.com/vil/virusSummary.asp?virus_k=99035

ID: A004
Title: Malicious code attack - intercept vote
Description: Malicious code intercepts the voter's vote.
Harm: Significantly impedes the voting process
Attackers: JRH, ET, SK
Visibility: High
Violations: Privacy
Likelihood: High

Preconditions: None
Triggers: None
Attack Flow of Events:
1. An attacker creates a virus which propagates through email.
2. The attacker releases the virus into the wild.
3. Voters release the virus by opening an email attachment.
4. The virus executes and runs itself in the background.
5. When the voter attempts to vote, the virus performs a "man-in-the-middle" attack. The virus intercepts transmissions between the voter and the vote server.
6. The virus then records who the voter is and what their vote was and posts it to a public newsgroup and/or web site.
Alternative Paths:
1a. other forms of malicious code could be used instead of viruses, such as malicious Java applets.
6a. The virus does not submit the voter's vote, but makes the user think the vote was submitted.
6b. The virus replaces the voter's vote with its own and submits it to the vote server.
Postconditions: The voter's vote has been compromised.
Special conditions: The user must go to a malicious website or execute viral code
Comments: This attack is guaranteed to occur
Defense Mechanisms: Educate users on the dangers of malicious code. Require voters to use anti-virus software. Require users to boot their computers from a "clean" boot disk.
Additional Information:
Gary McGraw and Greg Morrisett, Attacking Malicious Code: A Report to the Infosec Research Council, IEEE Software, September/October 2000.
http://dlib.computer.org/so/books/so2000/pdf/s5033.pdf

ID: A005
Title: Falsify ETA encryption key
Description: An Election Tabulation Authority falsifies their decryption key and prevents the results of the election from being tallied
Harm: High, voting results cannot be tabulated
Attackers: CETA
Visibility: High, but it is not apparent which ETA is the attacker
Violations: Completeness
Likelihood: High

Preconditions: None
Triggers: None
Attack Flow of Events:
1. Attacker receives key at beginning of election.
2. Attacker determines from exit polls or other methods that their preferred candidate will most likely lose.
3. Attacker falsifies their decryption key and the votes cannot be decrypted.
Alternative Paths: None
Postconditions: None
Special conditions: Attacker must be an ETA
Comments: In a traditional voting system, this could not happen.
Defense Mechanisms: Most likely a system will be used with uses an m-of-n decryption scheme, where only a certain number of the ETAs need to decrypt the results, and one corrupt ETA cannot disable tabulation.
Additional Information: Information on m-n encryption schemes can be found in [Sc94].

ID: A006
Title: Intercept votes coming into Voted Ballot Creator
Description: Votes coming into the VBC are intercepted and discarded
Harm: High, votes are not counted
Attackers: JRH, ET, CS
Visibility: Low
Violations: Soundness
Likelihood: Medium
Preconditions: None
Triggers: None
Attack Flow of Events:
1. Attacker has access to the network connection between the voter and the VBC
2. Attacker sees traffic between voter and VBC
3. Attacker intercepts vote going to VBC
4. Attacker discards votes
Alternative Paths:
3a,4a. Man in the middle attack could occur instead of vote discard
Postconditions: None
Special conditions: Must be able to gain access to network between voter and VBC
Comments: The connection between the voter and the VBC may be encrypted, so the attacker cannot replace the vote.

Defense Mechanisms: Encrypt communications, use strong verification for VBC, secure the network and monitor it
Additional Information: None

ID: A007
Title: Compromise vote server
Description: Use vulnerabilities in the vote server software to compromise the system
Harm: High
Attackers: JRH, ET, SK
Visibility: High or Low
Violations: Soundness
Likelihood: Medium
Preconditions: Vote server must have software vulnerability
Triggers: None.
Attack Flow of Events:
1. Attacker compromises software on vote server. This could either be the operating system, remote access software, or the vote server software.
2. The attacker shuts down the vote server to prevent voters from voting
Alternative Paths:
2a. The attacker submits false votes to the system.
Postconditions: None
Special conditions: None
Comments: There exist vulnerabilities in almost all software, so this is a very difficult thing to defend against
Defense Mechanisms: Software can be open to public review via open-sourcing it, network can be monitored to catch attacker attempting to compromise systems
Additional Information: More information can be found in [SMK01].

ID: A008
Title: Link digital signature to voter
Description: The voter is linked to their digital signature, compromising their privacy
Harm: High
Attackers: ET, JRK, CETA, CS
Visibility: Low or High
Violations: Soundness
Likelihood: Medium
Preconditions: Attacker must have direct access to the system

Triggers: None
Attack Flow of Events:
1. Attacker gets list of which voter digital signatures and voter names.
2. Attacker links a voter and a vote with the digital signature.
3. Attacker publishes the voter's vote to a website or newsgroup.
Alternative Paths:
3a. Voter extorts voter not to release their vote.
Postconditions: None
Special conditions: None
Comments: Because digital signatures are used to verify the voter, there is a stronger degree of authentication, but lower privacy guarantees
Defense Mechanisms: This is difficult to defend against. The best method for defense would be to have one group generate and distribute the digital signatures, and give a list of valid digital signatures to the vote server authorities. Alternately, and anonymizer could verify the digital signature and submit unsigned votes to the vote server. However, this introduces the additional problem of the anonymizer submitting its own fake votes.
Additional Information: Work on anonymizers can be found in [He97]

ID: A009
Title: Submit ballots with more than one choice for each office
Description: Voter submits a ballot with more than one choice selected for each office.
Harm: Low
Attackers: V, JRH, ET
Visibility: High
Violations: Unreusability
Likelihood: Low
Preconditions: None
Triggers: None
Attack Flow of Events:
1. Attacker selects more than one choice for an office.
2. Attacker submits ballot.
3. Extra votes are tallied.
Alternative Paths: None
Postconditions: None
Special conditions: None
Comments: This attack may be either accidental or intentional.
Defense Mechanisms: Vote server software should check to make sure that only one choice for each office is chosen using a zero knowledge proof.

Additional Information: [Ne00c] has information on the zero knowledge proof used to prevent this.

ID: A010
Title: Forge digital signature
Description: Attacker forges the digital signature on a submitted ballot.
Harm: High
Attackers: JRH, ET
Visibility: Low
Violations: Eligibility
Likelihood: High
Preconditions: None
Triggers: None
Attack Flow of Events:
1. Attacker forms ballot and signs with a fake digital signature.
2. Attacker submits ballot to vote server.
3. Vote is counted.
Alternative Paths: None
Postconditions: None
Special conditions: None
Comments: With strong cryptography, this is very difficult to do. However, it may still be possible with vulnerabilities not directly related to the cryptographic algorithms.
Defense Mechanisms: Vote server must verify digital signature is from a valid voter.
Additional Information: See [Sc94].

ID: A011
Title: Steal ETA encryption keys to get intermediate results
Description: Attacker steals the encryption keys of the ETA to get intermediate results of the voting.
Harm: High
Attackers: CS, ET
Visibility: High
Violations: Fairness
Likelihood: Low
Preconditions: None
Triggers: None
Attack Flow of Events:
1. Attacker steals the encryption keys of the ETAs.

2. Because votes are publicly posted so people can verify their vote was counted, the attacker has access to all encrypted ballots.
3. The attacker uses the encryption keys of the ETAs to decrypt the ballots and release intermediate results.
4. Voters who have not voted see intermediate results and are influenced.
Alternative Paths: None
Postconditions: None
Special conditions: None
Comments: None
Defense Mechanisms: Keep ETA keys safely guarded.
Additional Information: None

ID: A012
Title: Fake ballots substituted for real ones
Description: Fake ballots are substituted or added to the real ballots
Harm: High
Attackers: CETA, CS
Visibility: High
Violations: Verifiability
Likelihood: Medium
Preconditions: None
Triggers: None
Attack Flow of Events:
1. Attacker adds fake ballots to the real ballots
2. Ballots are tallied
Alternative Paths: None
Postconditions: None
Special conditions: Must have access to vote system and digital signatures.
Comments: None
Defense Mechanisms: Watch the system administrators closely.
Additional Information: None

ID: A013
Title: Voted Ballot creator submits fake votes to Vote Server
Description: The Voted Ballot Creator creates fake votes and submits them to the Vote Server
Harm: High
Attackers: CS, CETA
Visibility: Low or High
Violations: Verifiability

Likelihood: Medium
Preconditions: None
Triggers: None
Attack Flow of Events:
1. Attacker gets a list of valid digital signatures.
2. Attacker writes software which causes the Voted Ballot Creator to submit fake votes with valid digital signatures to the Vote Server.
Alternative Paths: None
Postconditions: None
Special conditions: Must have access to vote system and digital signatures.
Comments: None
Defense Mechanisms: Watch the system administrators closely.
Additional Information: None

ID: A014
Title: Web site spoofing
Description: An attacker creates a website with a similar URL to that of the real vote server.
Harm: Low
Attackers: JRH, ET, SK
Visibility: High
Violations: Reliability
Likelihood: High
Preconditions: None
Triggers: None
Attack Flow of Events:
1. Attacker registers a domain name similar to the actual vote server. In this case, if the server's URL was www.votehere.net, an attacker could register www.voteehre.net or www.voethere.net.
2. The attacker creates a "spoof" web site at their URL which looks like the vote here site.
3. The attacker's web site accepts digital signatures from voter and submits its own vote to the vote server.
Alternative Paths:
3a. The attacker's web site publishes the voter's choices to a public place.
Postconditions: None
Special conditions: Must register domain names. This may be prevented from happening.
Comments: The attack will definitely happen. Its impact will be very low, but it still adds confusion to the process.

Defense Mechanisms: The authentic web server can be authenticated with a digital certificate, but few users understand this concept. Votehere.net could also register all variations on their domain name.
Additional Information:
Web Spoofing: An Internet Con Game
http://www.cs.princeton.edu/sip/pub/spoofing.html

ID: A015
Title: Compromise DNS server
Description: Attacker compromises DNS server
Harm: High
Attackers: JRH, ET, SK
Visibility: High
Violations: Reliability
Likelihood: Low
Preconditions: None
Triggers: None
Attack Flow of Events:
1. Attacker compromises the DNS server serving the voter.
2. The Attacker shuts down the DNS server.
3. The voter cannot resolve the www.votehere.net domain name.
4. The voter cannot connect to the votehere.net website and cannot vote.
Alternative Paths:
2a. Attacker replaces the valid entry for www.votehere.net with a spoofed web address and performs an attack as in A014.
Postconditions: None
Special conditions: Must compromise DNS server
Comments: None
Defense Mechanisms: ISP for the voter must have redundant servers. URL for voting is given as numeric IP address instead of URL.
Additional Information:
CERT Advisory CA-1999-14 Multiple Vulnerabilities in BIND
http://www.cert.org/advisories/CA-1999-14.html

ID: A016
Title: Router compromise
Description: Attacker compromises router between voter and vote server
Harm: High
Attackers: JRH, ET
Visibility: High

Violations: Reliability
Likelihood: Low
Preconditions: None
Triggers: None
Attack Flow of Events:
1. Attacker compromises a router between the voter and the vote server.
2. The Attacker shuts down the router.
3. The voter cannot connect to the votehere.net website and cannot vote.
Alternative Paths:
2a. The attacker redirects traffic to a spoofed website as in A014.
Postconditions: None
Special conditions: None
Comments: Routers are often setup insecurely.
Defense Mechanisms: Close vulnerabilities in routers, use a secure routing information protocol
Additional Information: None

ID: A017
Title: Coercion
Description: Attacker coerces voter to vote a certain way
Harm: High
Attackers: UB, ET
Visibility: High
Violations: Privacy
Likelihood: High
Preconditions: None
Triggers: None
Attack Flow of Events:
1. Attacker constructively or destructively coerces the voter to vote a certain way.
2. Tally is affected. Alternative Paths: None
Postconditions: None
Special conditions: None
Comments: Because the voter does not have to be in a private location during voting, they can be coerced to vote a certain way.
Defense Mechanisms: Force users to go to designated polling stations, but this defeats the efficiency gains with ivoting.
Additional Information: See [BT94]

Appendix G

Attack Tree

- Completeness - All votes are counted correctly
  1.ETAs decrypt fake ballots
  2.Falsify ETA encryption key

- Soundness - The dishonest voter cannot disrupt voting

- Privacy - All votes must be secret
  1.Malicious code - records voters vote for further use
  2.Sniff and track votes coming into Voted Ballot Creator
  3.Substitute ballot, submit wrong one
  4.Compromise vote server
  5.Link Digital signature to voter
  6.Coercion

- Unreusability - No voter can vote twice
  1.Submit multiple ballots
  2.Submit ballots with more than one choice for each office

- Eligibility - No one who isn't allowed to vote can vote
  1.Digital signature captured upon distribution

- Fairness - Nothing must affect the voting
  1.Voter coerced into voting a certain way
  2.Malcode - replaces vote
  3.Steal ETA encryption keys to get intermediate results

- Verifiability - No one can falsify the result of voting
  1.Fake ballots substituted for real ones
  2.Voted Ballot creator submits fake votes to Vote Server

- Reliability - Nothing must prevent voters from voting
  1.DDoS against ISP
  2.DDoS attack against vote server
  3.Malicious code - shuts down client computer
  4.Corrupt DNS entry for vote server
  5.Shutdown DNS
  6.Compromise vote server
  7.Flood of voters crashes system