A Review of the EveryoneCounts Vote Casting Applet

Donald Newell

Senior Thesis, Department of Computer Science, Florida State University

June 1, 2008

Abstract

Modern day voting systems have come under much scrutiny in recent years. Whetherdue to a malfunction in an election, or through analysis by outside parties, errors in these systems are continually rising to the surface. The drive to meet consumer demand sometimes works in conflict with the need for thorough design and testing.

The internet provides unparalleled channels of communication to far reaching places. This characteristic, along with the internet's presence everywhere in modern countries, presents an attractive medium through which to use an electronic voting system.

Everyone Counts is an organization that has tried to implement a system whose appeal is two-fold: the utilization of the internet to allow voters to vote from the comfort of their homes, and the secure and private handling of the sensitive issues that are associated with voting in general. The vote casting applet used in the Everyone Counts (E1C) system achieves these two goals in many ways.

An improvement to the vote checking mechanism is proposed. This change would add a degree of verifiability that the vote was received-as-cast. The applet showed few weaknesses, however, because the scope of this review only encompassed the vote casting applet, an overall system review would need to be performed before a truly definitive opinion could be formed.

I. History of Cryptographic Voting Protocols

A. Introduction

This report documents our review of the electronic voting system knows as “Everyone Counts” or E1C. The E1C system is a cryptography-based system, which represents a new paradigm in elections. Because this system represents a dramatic departure from present voting systems in the United States, we begin our review by describing three well-known cryptographic voting systems.

B. Chaum / Neff's Scheme

There are many cryptographic voting systems documented in the literature. In this section, we highlight two seminal systems recommended by David Chaum[6] and Andrew Neff [3]. Chaum and Neff's protocols share the same structure: election initialization, ballot preparation, ballot tabulation, and election verification. Before the election, a set of election trustees are chosen that will participate in the election initialization phase by deciding on critical information to be used in the election process. The trustees must be from different groups, such that they have competing interests and are unlikely to collude. Ballot preparation begins when a voter arrives at the polling station and ends with the vote being cast. Each vote is cast in a private booth on a Direct Recording Electronic voting machine (DRE). The DRE then creates an electronic ballot that is sent to a public bulletin board, which is used as the "ballot box". Each ballot has a unique ballot sequence number (BSN) for use in auditing and verification, and has no information about how a person voted. The DRE prints a receipt that is used by the voter to confirm that the vote was cast, but the receipt cannot be used to determine the choices a voter made.

The next phase, ballot tabulation, is where each trustee completes certain stages of a publicly verifiable multi-stage mix net to anonymize the ballots. When the ballots enter the mix net, they are stripped of their BSN, eliminating any connection between the identity of the voter and the ballot that they cast. The mix net takes a set of encrypted ballots and performs decryption operations on them, randomly orders them, and produces plain-text ballots. Each trustee may provide a proof which can be publicly used to confirm there was no foul play in the process.

With such a complex system, it was important to clearly state the objectives of the protocol with regards to security. Each vote should be cast as intended. This means that the submitted ballot correctly represents the voter's choices. The submitted ballot should, in turn, be counted as cast, which is when the election results accurately represent the ballots cast. The two previously mentioned characteristics should be verifiable and duplicate voting should be prevented. Finally, a voter should not be able to prove to a third party which candidate(s) they voted for, which contributes to coercion resistance.

Candidate 1	10 10 01 10 01 01 01
Candidate 2	01 10 10 01 01 01 01
Candidate 3	00 00 11 11 11 11 00
Candidate 4	10 10 10 10 10 01 10

Ballot Matrix

Candidate 1	1X X0 X1 1X 0X X1 0X
Candidate 2	0X 1X X0 0X X1 X1 X1
Candidate 3	X0 0X X1 X1 X1 1X X0
Candidate 4	1X X0 1X X0 1X X1 X0

Opened Verifiable Choice

Figure 1

There are four main types of threats: malicious DRE's, malicious bulletin boards and trustees, coercive parties, and honest participants whose ignorance is taken advantage of by a malicious third party. Malicious DRE's could result from a rebel programmer or people that have access to the machines before the election. Malicious bulletin boards and trustees could destroy ballots or make them useless by destroying decryption keys. Coercive parties seek to find a means to verify a voter's choices, and incompetent participants can be taken advantage of in many parts of the process.

In Andrew Neff's scheme, the process starts with the trustees generating a master public key using a distributed key generation protocol. This key is used in the encryption of ballots. The basic interaction between the voter and the DRE consists of 5 steps: the voter chooses the candidate to vote for, the DRE encrypts the ballot, the DRE commits to ballot, unique BSN is assigned, and voter decides if they want a receipt. The DRE creates a verifiable choice (VC) after the voter

makes their selections. The VC is an nXl matrix of ballot mark pairs (BMP) (see figure 1). The number n is the number of candidates and l is the security parameter. Each BMP is a pair of encrypted O's or 1's that are encrypted using the master public key that the trustees created during election initialization. The BMP's for chosen candidate Ci will all be of the form (0,0) or (1,1) on row i. All unchosen candidates shall have rows filled with BMP's of form (0,1) or (1,0). Any departure from this standard indicates malfunctioning DRE's or foul play. Ideally, decrypted ballots will have 00 and 11 plaintext in the rows for selected candidates.

Verification of the VC consists of an interactive proof at the polling booth and comparing the voter's receipt to the bulletin board. The interactive proof provides a pledge bit p in the row of a chosen candidate and the voter chooses left or right bit and the DRE shows that the cipher text decrypts to p. The DRE also reveals the randomness used in the encryption. Because the voter can do this for l BMP's, it at least convinces the voter that their receipt accurately confirms their ballot's correctness. The average voter will not likely be able to complete the verification process without the assistance of a person or special software. This may be possible later using the receipt. The DRE prints the pledges on the receipt and then it prints the user's challenge bit string, where 0 means left element and 1 means right element.

The ballot station software submits the ballot to the bulletin board management server, which posts the ballot to the bulletin board. The voter's choice Ci is opened according to the challenge entered. The DRE randomly generates challenge bit strings for all of the unchosen candidates and half-opens them as well, so that the OVC is an nXl matrix of bits that does not reveal the voter's selection. The voter checks that the challenge on the receipt matches the challenge entered. They also make sure that the candidates names appear in the correct order. Also, the voter can check that the OVC is on the bulletin board and that the hash matches. To prevent coercion or vote-buying, the voter may be able to specify challenges for the unchosen candidates as well.

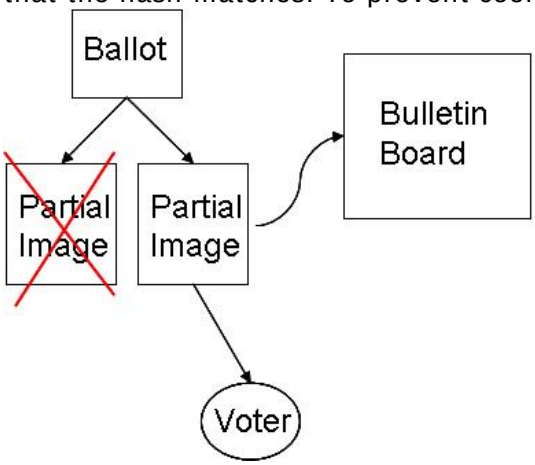

Figure 2 Chaum's Image Verification

A voter may choose between a basic or a detailed receipt. A detailed receipt contains the pledges, BSN, hash(VC), and challenges while the basic contains only the BSN and hash(VC). If a voter chooses a detailed receipt for a race, then they must choose a challenge and specify the unchosen challenges.

David Chaum's protocol is similar to Neff's, in principal, but differs in the verification implementation (see figure 2). Chaum's receipt is based on 2 layers of dot images that reveal the voter's choices when the 2 images are placed on top of each other. By themselves the images convey no information about the voter's ballot. The voter selects one of the layers to keep and that layer is printed on the receipt and sent to the bulletin board. The other layer is destroyed. The voter can verify that the 2 transparencies form the correct ballot image. Also, with either one of the transparencies and a trustee's key, the other image is recoverable. It is possible that an image that is slightly altered may pass the visual inspection of the voter. In that case, the DRE would not accept the fake transparency because it would not decrypt to the original ballot. In the ballot images, each pixel has a type, pseudo-random (P) or encrypted (E). In each transparency layer, the pixels are arranged in alternating order. Also, P pixels can only be on top of E pixels and vice versa. P pixels are generated using a pseudo-random stream that is based off of the trustee's keys and the BSN. E pixels are set so that the P pixel being put on top reveals a ballot pixel.

If there are several ways that a DRE can represent a voter's ballot, then the use of different formats may, in itself, be able to convey information. Since the bulletin board is public, a malicious individual could obtain information about how someone voted by the format that the ballot is displayed. A ballot that stores information about the voter in a subliminal channel affects privacy and enables vote coercion. This could be done through storing the time a vote was cast and a malicious observer recording the times voters leave a voting booth. A poll worker obtaining a voter's BSN has similar implications. The basic threat model is a malicious DRE colluding with an external party. This subliminal channel comes about as a result of the bulletin board being public.

Some of the cryptographic operations in Neff's scheme use random values. If the DRE chooses these values, then there is the possibility of a subliminal channel. A DRE may hide a bit, b, in the encryption of a message by encrypting the message, using a new random number each time, until the least significant bit of the encrypted message is b.[3] In Neff's scheme, the BMP uses El Gamal cipher text pairs where the

encryption is randomized. The Opened Verifiable Choice (OVC) shows half of the random values (see figure 1). The random encryption parameter w is revealed when one of the BMP are opened. If the DRE maliciously uses the same w for both of the pairs, then information can be stored in w. Much information can be conveyed through this attack. One way to deal with the randomness attacks is by designing protocols that don't rely on randomness. Chaum's protocols use deterministic primitives. Neff proposes the use of tapes that contain predetermined random numbers which the DRE would have to use. Then it would just need to be verified that the DRE actually used the bit from the tape. One way is if commitments were posted to the bulletin board, such that they could be compared to the values used by particular DRE's.[3] Also, trusted hardware was proposed as a possibility to supply the random bits, but the need to verify and monitor the hardware and software seems less desirable.

The ability to represent a voter's choice in multiple ways allows the DRE the ability to embed information in the particular configuration it chooses. Chaum's use of ballot images allows subliminal information to be stored in small modifications to the pixels.[3] Such a semantic channel is only useful after the mix net, since the information is stored in the plaintext ballot, whereas the random channels can be utilized immediately on the public bulletin board. One way of dealing with this attack is the use of standard ballot formats[3]. Any differ from the format would be kept off of the bulletin board, because one ballot could convey information compromising all of the ballots cast at a particular DRE. The order in which ballots appear on the bulletin board would also need to be standardized. Subliminal channels are a serious threat, because they bypass all security mechanisms in place.

It is generally accepted that humans unfamiliar with cryptography are likely to make mistakes and miss minor deviations when using it. Malicious parties can take advantage of this in a number of ways. A DRE could reorder the steps in Neff's scheme to find out if the voter wants a basic or detailed receipt. If the user chooses basic, then the DRE can choose whatever ballot it wants to send, since the receipt only prints the BSN and a hash of the VC. Also, if the DRE can reorder the steps of the protocol in such a way that the user tells the DRE the challenges they want to use before the DRE prints anything on the receipt, then the DRE can create a ballot for any candidate and place the challenge in the position that the voter expects. In Chaum's scheme the DRE could determine what transparency the voter wants, and then generate an opposing image to create a different ballot, meanwhile the voter sees the correct transparency.

These attacks capitalize on ignorance or inattentiveness by the voter. Apathetic voters could enable a malicious poll worker to observe discarded receipts and use this information to tell a malicious DRE which ballots are ok to alter or delete, since they are not likely to be checked on the bulletin board.

There is also a danger that the DRE may perform operations that the voter is aware of, but that the voter cannot prove to a third party. Such attacks are: using invalid signatures to make receipts appear forged, printing the wrong machine id so as to deflect attention or even ignoring voter input all together. These attacks could be mitigated in a number of ways. Random auditing would cut down on the ability of DRE's to behave abnormally, and voter education stressing the importance of keeping receipts would also help. Other methods include using preprinted receipt paper and the use of trusted hardware to verify the DRE signature.

Sometimes detection of an attack is easy, but the recovery is difficult. This is the case with some of the Denial of Service (DoS) attacks that are possible on this system. Ballot duplication is where a malicious DRE submits multiple ballots with a valid BSN. This is easy to detect, but we do not know which ones are valid. Another attack is ballot stealing, where a DRE submits a ballot with its own choices and a valid BSN while printing an invalid BSN on the voter's receipt. Perhaps the most serious attack is when a malicious bulletin board erases a trustee's key, thus rendering the encrypted ballots useless. Selective DoS is when a malicious DRE or individual triggers a DoS attack to support a candidate that is losing in a particular precinct. The simplest and most impractical solution would be to allow all of the cheated voters to re-vote, or to redo the entire election. A much more appealing mitigation strategy is the use of a Voter Verified Paper Audit Trail (VVPAT). VVPAT is a paper record that a voter could verify before the ballot is cast, and that is placed into a ballot box for use in auditing or recounts. This seems to be the best solution, but it is not without its downfalls which are left to be discussed another time.

Under-specification is an issue with regards to several different components in these protocols. The bulletin board is a major part of the system. The data storage must be stored securely and free from mechanical failure. Also, every viewer should see the same copy of the bulletin board. It is possible that the bulletin board could display different versions to voters based on what the voter is expecting to see, while the real state of the election is hidden. It is also suggested that the architecture of the bulletin board be specified by both Neff and Chaum. The assignment of BSN's through smartcard use or counter

implementations would counteract several of the previously mentioned attacks as well. Several of the attacks manipulated what appeared on the DRE's screen to trick the voter. Consequently, specification of the user interface would be helpful as well. Finally, the tallying software is a key piece of the puzzle. If all of the trustees use the same software then a single programmer could have wide-reaching impact, whereas if all of the trustees use different software, interfacing becomes an issue. Neither Neff nor Chaum specify the tallying software. [3]

After evaluation of these 2 protocols, there are several questions that arise and deserve further investigation. The presence of subliminal channels in cryptographic protocols would be a valuable problem to address, as well as the security model for the mix net. Perhaps the most difficult issue is how to deal with the human factor in the protocols. With the wide range of abilities that voter's possess, how should voter education be structured to produce the most effectiveness? Overall, Neff's and Chaum's protocols are an improvement over many other voting systems because of the ability to verify the record of the vote on the bulletin board while still maintaining coercion resistance. The presence of weaknesses is not totally disconcerting since there are a variety of ways to mitigate their danger. The main issues to be confronted are that ballots on the bulletin board must be unique and there needs to be an organized method of mitigating the problem of the human component.

C. Prêt a Voter

The Prêt a Voter scheme presents a simpler approach to voting, when compared to Chaum and Neff's schemes. In Prêt a Voter, ballots consist of a sheet which can be separated into left and right-hand pieces. A randomly ordered candidate list is placed on the left, while a voting grid for the user's choice is placed on the right side. At the bottom of the right-hand strip is a random-looking code that is used during tabulation to determine the left hand candidate order. After marking the appropriate place on the right strip, the voter separates the 2 sides and destroys the left side. The right side is then scanned, the code at the bottom is recorded, and the voter keeps the right strip as a receipt. The votes are posted to a WBB so that voters can see that their receipt is stored correctly. When the votes are counted, a group of tellers, acting together, decrypt the ordering of the candidates using their individual keys. Three areas of concern are: incorrect cryptographic values on the right strip, transmission errors of the receipt to the WBB, and teller errors during decryption [5].

Chaum's and Neff's protocols had significant concerns in the area of subliminal channels, where Prêt a Voter avoids such problems by design. The ballot reader never learns the voter's choice, so it has no valuable information to convey. As long as the left strip is destroyed before the right side is scanned, it is unlikely the reader could obtain any information. Random subliminal channels are not a problem since the cryptographic operations take place when the forms are created, long before a voter makes a choice. Semantic subliminal channels are addressed through the way a vote is recorded as a cell number and the cryptographic number (onion) [5]. Ballot forms could be a subliminal channel, but there is no information that would be valuable to convey in this way. As in Chaum's scheme random partial checking is used to prevent collusion between the tellers. A key part of the security of the system is the fact that the authorities can commit to the cryptographic material before the election and the forms could be randomly audited for well-formedness.

A viable threat could be the posting of duplicate ballots or the posting of ballots with invalid onions. There also could be significant damage if a failure occurs during the mix net that destroys a large number of ballots. A possibility is to replace decryption with re-encryption. Separating the mix and decryption would simplify key management because the mix tellers would not possess private keys. A failed mix teller could easily be replaced. This has the added benefit that the mix and audit could be independently re-run. This approach still does not protect against the corruption of the decryption tellers' key, but threshold encryption schemes have been proposed as way of mitigating this threat.

The issue of recovery from an attack could be partially solved through the use of a VVPAT. However, VVPAT presents issues of voter privacy. Encrypted receipts may prevent this problem. Also, the use of discarded receipts could be addressed by auditors randomly checking the WBB using the VVPAT, thus eroding confidence that discarded receipts represent unchecked ballots. There should also be machines available to the voters, that verify the signature on their receipt, in order to prevent a DRE from printing invalid signatures to discredit a voter. Finally, as stated before, a robust storage system is necessary for the WBB.

The Doll Matching attack [3] in Chaum's scheme is when a voter is unable to catch changes to the dolls through visual inspection. The use of bar codes is suggested in [3] to address this issue. If the bar codes do not line up then the voter would know that the vote has been tampered with. This attack is not applicable to Neff's scheme or Prêt a Voter.

An attack that is specific to Prêt a Voter is chain voting [5]. This occurs when a malicious individual obtains a form outside of the polling station. The individual then catches a voter before they enter the polling station and tells them to cast that ballot for a particular candidate and emerge with a new ballot. This approach is effective because the forms are a controlled resource. The coercer's power comes from their knowledge of the association between the onion and a particular candidate order. To counteract this, a scratch-strip could be placed over the onion until an official observes it is intact and oversees its removal when casting the ballot [5]. A method of scanning scratch-strips with a laser to obtain the information beneath it has been suggested as a means of circumventing this measure and needs to be addressed. The use of the scratch-strip also mitigates the threat of double-voting.

The knowledge of the correlation between candidate orderings and their respective onions is the main information that needs protection. One approach is that no single authority should have access to all onions and candidate lists, because this information could be leaked in a number of ways. Also, the correctness of the said correlation on the ballots is protected by the fact that the authorities commit to it and it is randomly audited. Distributed ballot form construction is one way to achieve this protection. A group of ballot clerks perform an encryption mix on a large set of El Gamal onions, producing a permuted, random set of onions. The last clerk produces 2 re-encryptions for each onion. Each onion pair is then printed on the ballots, one on each side. The right onion is covered with a scratch strip and the left onions are sent to the tellers, who send back candidate lists that correspond to the left-hand onion. The lists are printed on the forms and the left onion is destroyed, leaving only the covered right onion. Random audits of the process could be performed to ensure its correctness to an acceptable degree.

The last avenue that this correlation could be obtained is through a voter retaining the left strip of the ballot after they leave the polling station. Thus, the enforced destruction of the left strip is a necessary step. This could be achieved in a number of ways which include: the destruction of the strip in the presence of an election official, the availability of decoy left hand strips, and the candidate lists being printed on a scratch strip that covers the onion.

A voter may verify the ordering of their ballot by submitting the onion to the tellers. The tellers then return the associated ordering of candidates. It must be the case that, once a form's order is checked, that the form is prevented from being used again. Since the onion/ordering association is known, a malicious coercer could verify a person voted correctly on the WBB using that ballot. The previously mentioned scratch-strip implementation would prevent this from happening.

The Prêt a Voter scheme takes a different approach than the protocols developed by Chaum and Neff. The use of "dumb" DRE’s is useful in avoiding the problems with subliminal channels seen in the other two schemes. However, it is vital that the threat present in chain voting is addressed in order for the system to be successful. The simplified approach seen in Prêt a Voter is desirable and shows promise as a viable secure voting scheme.

II. The E1C system

Before addressing E1C, we emphasize that this review is an academic exercise. An important component of this review is discovery of voting system technology. Thus, this report should not be seen as a definitive system or security review of E1C, but rather provides only our review insights in the overall discovery process.

The previous systems discussed all depend on human interaction with a highly customized and regulated computer in a very organized and secure environment. A poll worker can physically verify that the person is who they say they are by looking at picture identification and by verifying highly personal knowledge that the person has. All of this is possible because of an assumption that voters can physically travel to a central polling station to vote. These do not deal with voters whom are disabled or may be overseas and still would like to cast their vote. In the United States, these individuals cast paper "absentee" ballots. Website voting systems provide a new absentee ballot solution.

The Everyone Counts solution is to use a java applet as the vote casting mechanism. The applet is downloaded to the voter's computer and the selection portion of the voting process can be run with the internet disconnected, so as to deter monitoring by an outside party.

Along with the applet come the appropriate ballots for the voter's district and the certificates for

the transmission of the vote. The submission of votes to a centralized server creates vulnerabilities to Denial-of-Service attacks for website voting. E1C attempts to deal with this by offering a version of the E1C system that uses a Peer-To-Peer (P2P) network as the backbone of the vote submission process (see Figure 5). A vote is encrypted by the applet and then submitted to a distributed network, and it may even be submitted multiple times to achieve redundancy. The applet then provides a receipt to the voter that can be used with a vote verification service at a later time to ensure that the vote was cast, captured, and counted correctly. After votes are sent to the network, several private "receiver" servers are used to poll the network for votes and download new votes found. After votes are downloaded from the network, they are then sent to a central tabulating server to be recorded.

III. Overview of Applet functionality

The main focus of this project was reviewing the applet source for possible weaknesses and verification of aforementioned security mechanisms. When the applet is downloaded from the server, it is customized with the appropriate candidate lists and certificates for the voter who is receiving it. The program either has the voter enter their credentials (voter id, personal id, date of birth), or these values are already loaded through a prior credential acquisition. Then, the applet takes the voter through a series of screens where they vote for particular candidates by using the graphical interface, until all voter selections are complete. Then the software hashes the user credentials, including the password, and uses this to create a receipt. The receipt is then encrypted with the vote and put on the network. The network sends a response back indicating success or failure. Upon success the user obtains the receipt. This is where the job of the applet is finished and the server side of the process takes over.

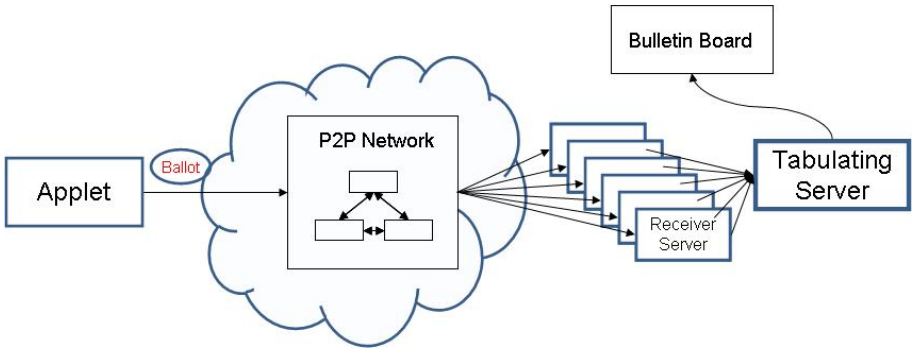

Figure 3 - P2P version of the E1C system

The voter can be confident that the process worked by checking one of several verification sites. The password that the voter entered into the applet was hashed with the credentials and encrypted with the vote. This means that if the receipt shown on the website matches the receipt that the voter possesses then there can be confidence that the vote was decrypted correctly by the server. Since the applet is prepackaged and signed, there is sufficient confidence that the voter will receive an unaltered copy of the applet with the correct certificates. The voter has the option of completing the ballots while the computer is not connected to the internet. This would be desirable since there is a possibility for surveillance by a third party if the computer is connected. This leaves the cryptographic operations and the communication with the server as the main areas of concern.

A. Initialization of the Applet

An important aspect of using the applet is that the different ballots can be generated before the election. After the voter's credentials are authenticated by the applet download site, a jar file is sent to the voter that contains the Election Returning Official's (ERO) certificate and all of the ballots, in addition to the applet. The details on how exactly the user is authenticated and the layout of the download page has not been provided for this review, so no critique is provided here. However, it should be noted that this is an important part of the system. The distribution of the credentials used to log in needs to be reasonably secure. In past implementations of this system, pins were distributed through paper mail with a high degree of success. This instance, however had all of the voter's in the same country, whereas the same security may not be achievable in a distributed environment where voters are spread over different countries, much less, continents.

When the applet is first opened, it either displays an authorization page or goes straight to the ballot, depending on whether or not the pin and date of birth (dob) values are already set. When the voter is authenticated, the server informs the applet which ballots to render. [7] After the voter has been autenticated, they can disconnect from the internet. This would reduce the threat posed to activity on the internet. The data structure for creating and storing the vote is created, and the public key is imported from the certificate that is in the jar file. Then the panels for displaying the ballots are generated, along with the panel for entering the user's password for use in the receipt. At this point, the interface has been created and is ready for input from the user.

B. Completing the Ballot

The program flow of the applet is directed by a set of states and button clicks. The general idea is that the voter will complete a page and then click next, previous, or undo and depending on what state the applet is in currently, the applet will display the next page. If the authorization page is displayed, then the voter must enter their credentials and the applet will do some basic credential checks (not null, check digit). If these checks pass then the applet sends a hash of the salt (a predefined number hard coded into the applet), voter identification number (vin), personal identification number (pin), and date of birth (dob) to the server for authentication (see figure 4). The server replies with a string message that confirms success if the hash checks out. If anything else is returned then an error message is generated.

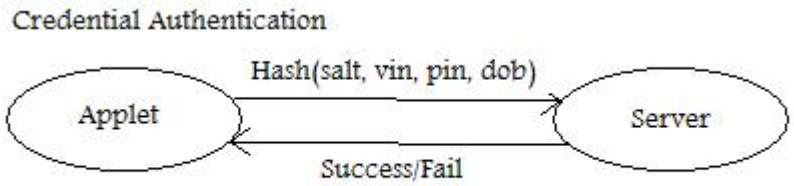

Figure 4 Applet Credential Authentication Protocol

If no error is encountered, then the applet displays the first ballot. From here on the user simply selects the appropriate candidates and clicks the next button. This is repeated until all of the ballots have been completed, at which point the ballot is ready to be prepared for transmission to the server.

C. Casting the Vote

After the voter has filled out all of the appropriate ballots, the next step is to prepare the vote to be sent to the server. First, the voter submits a password to the applet that is used in the later parts of the system to provide assurance that their vote was decrypted correctly. Then the states from all of the check boxes are recorded in a single string, which is called "flattening" the vote. After this, the applet hashes the salt, vin, pin, and dob, which is called the "base". The salt, vin, pin, and user-supplied password are also hashed

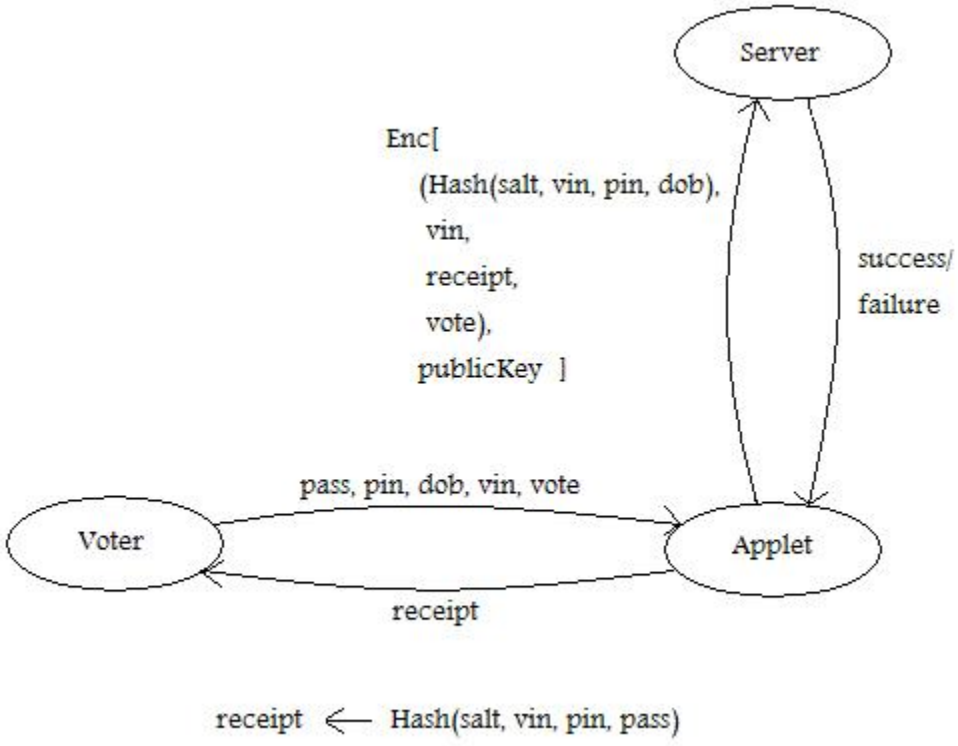

Figure 5 E1C Applet vote submission protocol

and this serves as the receipt. The applet then encrypts the base, vin, receipt, and flattened vote using the public key acquired earlier from the certificate, and sends this to the server.

The URL for the server is pre-loaded into the applet during initialization. Upon receipt of the encrypted vote object, the server stores the vote and signals the applet. Upon success the user is given either a simple or detailed receipt, and the applet closes. Decryption of the vote object and authentication of the voter credentials occurs offline at a later time to reduce reliance on the server. [7] The vote casting process is shown in Figure 5. Because the receipt is a hash of the credentials and the password that the user entered, the user can go to a receipt verification website at a later time and see that the same value is posted as is on their receipt. Since the receipt was encrypted with the vote string, if the server possesses the correct receipt, this means that, with a high probability, the server received the vote message that was sent by the voter. Also, since the voter will input personal credentials, such as vin, to look up the receipt, a correct value for the receipt also implies that the vin was correctly decrypted and also associated with the correct receipt. These two facts help to build confidence that the vote message was received unaltered.

IV. Findings

We reiterate that the purpose of our review was to identify E1C system properties. In doing so, we noticed two potential weaknesses, which we report in this section.

A. Server Identity Unverified

When the applet is sending the encrypted vote to the servers, it uses a predefined url with an appended instance id. After sending the vote, the output stream is closed to the server and an input stream is opened. The program then acquires a response from the server that is a boolean value representing successful transmission of the vote or not. Nowhere during this process does the applet verify the identity of the server. The same problem occurs in the initial stages of voting where the applet sends a hash of the user credentials to the server to authenticate the voter. This leaves the user vulnerable to a man-in-the-middle attack where a malicious individual may act as the server and send success messages back to the applets, meanwhile they may drop all votes.

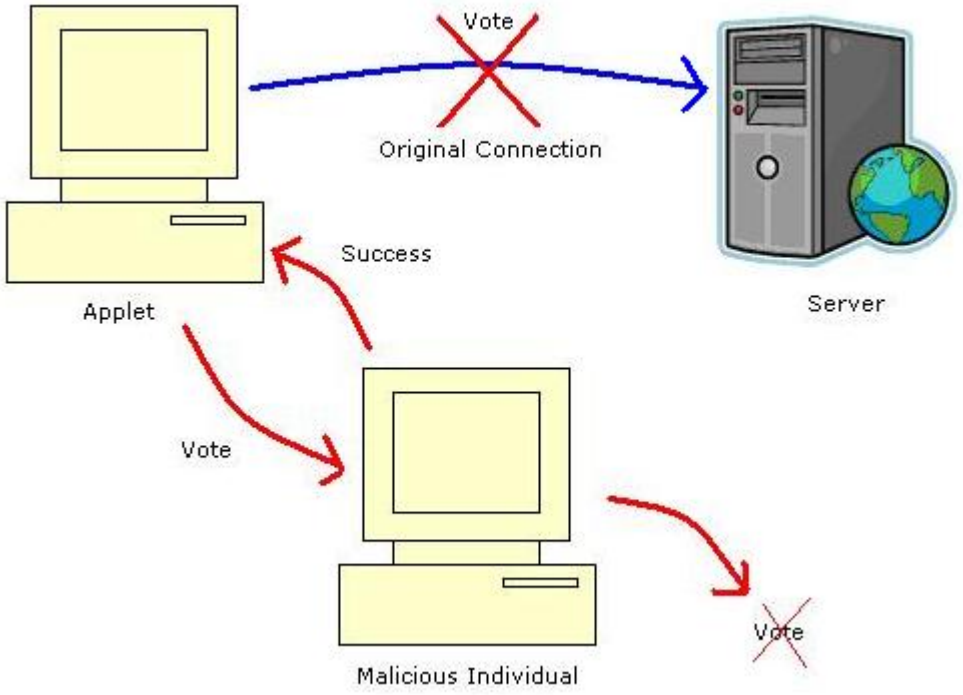

Figure 6 MITM Attack

This kind of attack would only be effective if a fairly large concentration of voters were present in one location. One possible scenario is an overseas military base. If a malicious individual were sniffing packets going to and from a military base during an election, it may be possible for such an individual to simply drop all outgoing packets that are identified as credential checks and return any random value and thus the credential authenticator would fail. This attack, however, would not be as effective as the previous attack because in the first, the voter doesn't ever know that anything went wrong.

B. E1C Strengths.

All of the E1C cryptographic operations were completed using a third party package called BouncyCastle [4]. This package is widely used and is licensed under the MIT license. The applet is downloaded in a signed jar file that contains the public key for the ERO. The jar file is signed, which prevents tampering. The certificates are loaded directly into BouncyCastle Data Structures. All sensitive information is hashed and encrypted before it is sent to the server.

Buffer overflows are a continual source of problems in applications written in C and C++. The choice of using java was also beneficial in that the risk of memory leaks and buffer overflows is mitigated through the use of garbage collection. In past reviews, the main problems arise from trying to assure the voter that the vote was "recorded-as-cast". Neff's scheme attempts to do this through the use of pledge bits and the OVC showing actual bits from the vote matrix. As stated previously, this opens up the vulnerability of subliminal channels. This vulnerability is not present in the E1C system because there is no displaying of information about the vote on the bulletin board.

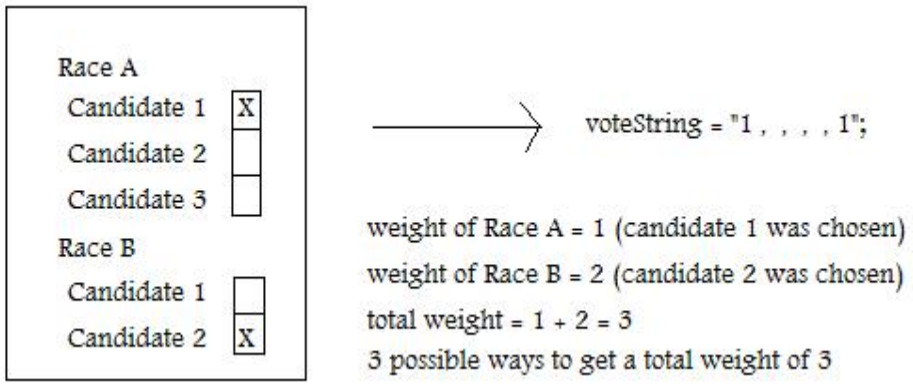

Figure 7 Proposed Improvement

C. Possible Improvements

A second possible weakness in the E1C scheme is in the use of the receipt. The receipt is a hash of the salt, vin, pin, and password. Although the correct display of this value provides a reasonable amount of confidence in the correct receipt of the vote, it could be stronger. The vote string is represented as a comma separated vector (csv). This means that a vote is designated by a '1' placed in between the corresponding set of commas, e.g. 1,,,,1,,1. For each race in an election there are n candidates. If, for each race, the candidates were numbered 1 through n, this number could be called $\text{weight}_{\text{candidate}}$ and used as an integrity check. If the weight of the chosen candidate for each race was summed, a $\text{weight}_{\text{total}}$ would be obtained. For any race that the voter chooses not to vote in, a weight of zero would be used. This process is seen in Figure 7. Take this total weight and hash it with the hash of the salt, vin, pin, and password and give it to the user. Nothing needs to be changed about the vote message that is sent to the server. It should still be the same as in Figure 5. What is different is that the server will decrypt the message, calculate the total weight of the vote string, and

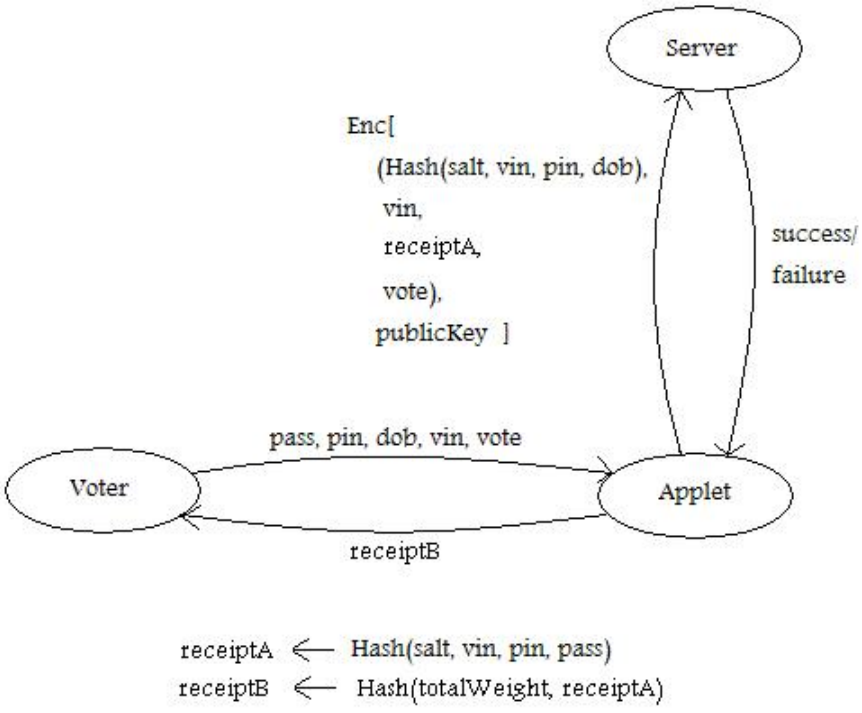

Figure 8 Improvement Protocol

hash the total weight with the hash of the salt, vin, pin, and password that was sent with the vote string (see Figure 8). This new receipt can then be posted to the vote checking website. This now provides the voter with increased confidence in the system if the numbers match. It also still prevents the voter from proving how they voted because several different ballot voting patterns would add up to equal the same number. Even further security could be added by having several different candidate orderings on the ballots for each race.

V. Conclusion

The system created by EveryoneCounts is well designed. The measures that were taken to ensure the security of the applet and the privacy of the voter are sound. The convenience of being able to download an applet in the privacy of your home is undeniable, and the confidence that is built through the vote checker website has the right goal in mind.

We found very few problems with the EveryoneCounts applet. The only detected weaknesses are related to the inherent problem of using the internet as the backbone for any system. Denial-of-Service and man-in-the-middle attacks are potential pitfalls. However, the man-in-the-middle attack requires a densely populated area of voters, which will greatly depend on the particular election.

We proposed a possible improvement to the vote checking system. The use of a vote weight is an integrity tool to verify the receipt of the vote as it was cast.

VI. Bibliography

[1] *A Virtual Private Network for Internet Voting,* Everyone Counts PL, January 2004

[2] Craig Burton, Shanika Karunasekera, Aaron Harwood, Duana Stanley, and Ioanna Ioannou, "A Distributed Network Architecture for Robust Internet Voting Systems", Everyone Counts, Department of Computer Science and Software Engineering, University of Melbourne, Australia 2005.

[3] Chris Karlof, Naveen Sastry, David Wagner, "Cryptographic Voting Protocols: A Systems Perspective", University of California, Berkeley.

[4] www.bouncycastle.org

[5] Peter Y. A. Ryan and Thea Peacock, "Pret a Voter: a Systems Perspective", University of Newcastle, 2005

[6] D. Chaum, P. Ryan and S. Schneider. A practical voter-verifiable election scheme. In *Proceedings of the 10th European Symposium on Research in Computer Security (ESORICS 2005)*, pages 118-139, 2005.

[7] Burton, Craig, Chief Technology Officer, EveryOneCounts.com, Personal Communication, May 2008.