Towards a Practical, Secure, and Very Large Scale Online Election

Jared Karro and Jie Wang
Division of Computer Science
The University of North Carolina at Greensboro
Greensboro, NC 27402, USA
Email: {jqkarro, wang}@uncg.edu

Abstract

We propose in this paper a practical and secure electronic voting protocol for large-scale online elections. Our protocol satisfies a large set of important criteria that has never been put together in a single protocol before. Among all electronic voting schemes in the literature, Sensus, a security-conscious electronic voting protocol proposed by Cranor and Cytron [CC97], satisfies the most of what we desire. Sensus has been implemented and used in mock elections. However, Sensus suffers from several major drawbacks. For instance, we show that even if all voters follow the Sensus protocol honestly, some voters' votes may still be replaced with different votes without being detected. Our protocol overcomes these drawbacks.

1. Introduction

Democratic societies are founded on the principle of elections. However, it is not unusual that many eligible voters in a democratic society do not participate in elections. One of the common reasons for not participating is that voters find it inconvenient to go to the polls. In conventional elections, voters must go to a designated location near their residence. However, for various reasons voters are not always able to make it to these locations. They may be out of town on work or on vacation. Even if they are in town, their daily schedule may not permit them to get to the ballots.

With the rapid growth of the Internet, specifically the World Wide Web, voting online provides a reasonable alternative and in the future may replace conventional elections. Voting online would allow voters to participate in an election in any location that provides Internet access. Voters could cast their ballots while at work, at school, or in the comfort of their own home. Many public libraries have computers with Internet access that could also be used in elections. In some places, bookstores and coffee bars are also starting to provide Internet access. For those voters still without Internet access, voting districts would still have designated locations, only computers, instead of voting booths, would be used. There would be no need to limit voters to a district.

The idea of electronic election over computer networks has been studied intensively for over fifteen years. A variety of cryptographic voting protocols have been proposed to minimize election fraud and maximize voter privacy (for example, see [Be87, BT94, Ch88b, Co86, CF85, C+96, CGS, CC97, F+93, IV91, MV98, NS91, NS, N+91, Sal96, Sch96, SK94]). Most of the early-proposed protocols only deal with a few certain issues of elections, mostly for theoretical interests. As pointed out in [F+93] and [CC97], such protocols are impractical to implement for a large-scale geographically distributed voting district. For a survey of several such protocols we refer the reader to Section 3.2 in Cranor and Cytron's paper [CC97]. So far there has not been a single government election done over the Internet.

Fujioka, Okamoto, and Ohta [F+93] studied how to make online elections practical and proposed a voting protocol using cryptographic techniques of blind signatures and anonymous communication channels. Their protocol also uses central facilities to administrate elections and count votes. They justified that using central facilities is necessary for a voting scheme to be practical. Built on this work, Cranor and Cytron [CC97] recently designed and implemented a security-conscious polling protocol, called Sensus. However, Fujioka et al.'s protocol and the Sensus protocol suffer from several major drawbacks (we will describe these drawbacks in Section 3). Some of these drawbacks are due to the use of blind signatures in large scales and the unpractical assumption of using anonymous communication channels (note that CPU identification numbers have been embedded into the new Intel's Pentium III chips that can be broadcast over the Internet). These drawbacks hinder Sensus from being used in large-scale elections.

To design a voting protocol that is secure and usable in large-scale elections, it is necessary and important to identify a set of criteria to help achieve our goals. We will describe these criteria in Section 2. The rest of the paper is organized as follows. In Section 3, we will discuss several major drawbacks of the Fujioka-Okamaoto-Ohta voting protocol and the Sensus protocol. In Section 4, we will present a new design of an electronic voting protocol to overcome these drawbacks. Our protocol also uses central facilities, but it does not use blind signatures or

anonymous communication channels. Moreover, our protocol can be used in elections where we wish to know who has voted and who has not. This property is desirable in Australian elections [MV98]. In Section 5, we discuss security measures and implementation issues of our protocol. In Section 6, we provide proofs that our protocol satisfies all the criteria defined in Section 2 and overcomes the drawbacks of the Sensus protocol. In Section 7, we outline a number of additional properties of our protocol.

2. System requirements

A good electronic voting system should not sacrifice voter privacy or introduce opportunities for fraud. For an electronic voting system to be useful and acceptable by voters, it must be at least as secure as conventional voting systems. We use the following set of nine criteria to ensure that an electronic voting system is secure and practical for large-scale elections.

Democracy. Only eligible voters are permitted to vote, and they can do so only once.

Accuracy. A voter's vote cannot be altered, duplicated, or removed without being detected. Invalid votes are not tabulated in the final tally.

Privacy. Votes remain anonymous.

Verifiability. Voters can be sure that their votes are tabulated correctly, but voters are not required to verify their votes in order to ensure election integrity.

Simplicity. Voters can finish voting quickly, with minimal equipment or special skills.

Mobility. Voters are not restricted to physical location from which they can cast their votes

Efficiency. The election can be held in a timely manner *(i.e. all computations during the election are done in a reasonable amount of time and voters are not required to wait on other voters to complete the process)*.

Scalability. The size of the election will not drastically affect performance.

Responsibility. Eligible voters who have not voted can be identified. (This is an optional requirement.)

Among these criteria, democracy, accuracy, privacy, verifiability, simplicity, and mobility are directly relevant to the voters, which are adapted from [CC97]. The criteria of efficiency, scalability, and responsibility are added to our system.

For the privacy criterion, we may further require that no voter can prove that he or she voted in a particular way to prevent vote buying and extortion. But as pointed out in [CC97], unless voters are required to cast their votes from inside a solitary voting booth, voters will be able to prove how they voted by allowing buyers to observe them while they are casting their votes. This requirement would comprise mobility, one of the major reasons to hold an online election.

The current US government elections do not satisfy the verifiability criterion. If an election booth has malfunctions, for example, then some voters' ballots may not be counted correctly and the voters are not able to detect it. In the past, elections have also been held in which ineligible voters, even the deceased, have been allowed to cast their ballot.

Conventional election systems also do not handle mobility easily. For those voters who will not be in their home districts during the election and wish to vote, they must file absentee ballots. But due to time constraints, this may not always be possible, as their absence may not be known until the last minute.

The criteria of simplicity, efficiency, and scalability imply that in such a voting system, voters cannot be required, or expected to communicate with other voters; and voters cannot be required to do all the computations of the election. This means that some central facilities must be employed by the system.

The responsibility criterion is not required in US elections, but it is required in Australian elections. By Australian laws, eligible voters are required to participate in government elections; they are subject to punishment if they do not participate without acceptable reasons [MV98].

3. Voting with blind signatures and anonymous communication channels

Many electronic voting protocols have been proposed during the past fifteen years as we mentioned in Section 1, but none of them seem to fit our set of requirements as nearly as Sensus. Many of these protocols, while of theoretical interest, are not practical to implement for a large number of geographically distributed voters [CC97]. Sensus, on the other hand, has actually been implemented and used in mock elections. Sensus is based on the voting protocol proposed in [F+93], which uses blind signatures and anonymous communication channels to administrate elections. In this section we will first outline these two protocols. We will then show that these two protocols suffer from several major drawbacks.

We begin with Fujioka et al.'s protocol [F+93], which consists of voters and three central facilities called *registrar*, *validator*, and *tallier*. Note that in [F+93], the validator is called the administrator and the tallier is called the counter. The registrar compiles a list of eligible voters, which could be performed before the actual election begins. (We note that the registrar facility is not mentioned explicitly in [F+93].) The protocol consists of seven phases outlined below, where the registration phase, not included in [F+93], is added here for completeness as in the Sensus protocol.

Registration phase. The registrar compiles a list of eligible voters prior to an election. Eligible voters generate public/private key pairs for signing ballots, and register to vote by sending the registrar their voter identifications and the public keys, which are placed in a registered voter list. (See [CC97] for a detailed implementation of this phase.) The registrar then sends the list to the validator.

Preparation phase. The voter *V* prepares a voted ballot *b*, encrypt it with a random string *k* he/she selects as in the bit-commitment scheme [Na90]. Assume that the committed ballot is *x*. The voter then blinds *x* into a new string *e*, signs *e* into a new string *s*, and sends (*I*, *e*, *s*) to the validator, where *I* is *V*'s ID.

Authorization phase. Using the registered voter list, the validator verifies that the signature *s* belongs to a registered voter *I* who has not yet voted, signs the ballot *e* into a new string *d*, and returns *d* to the voter.

Voting phase. The voter *V* retrieves the blinding encryption layer, revealing an encrypted ballot *y* signed by the validator, and sends the pair (*x*, *y*) to the tallier via an anonymous communication channel as described in [Ch81, Ch88a, Pf84].

Collecting phase. The tallier checks the signature *y*, using the validator's public key, to make sure that *x* is from a legitimate voter, and places (*x*, *y*) on a list of valid ballots.

Opening phase. At the end of the election, the validator publishes the number of voters who were given the administrator's signature, and publishes a list of all triples (*I*, *e*, *s*) it has received; and the tallier publishes the list of valid ballots. The voter *V* then checks that the length of the list is equal to the number of voters, and that his/her vote (*x*, *y*) appears on the tallier's list, with index *n*. The voter then sends (*n*, *k*) to the tallier via an anonymous communication channel.

Counting phase. The tallier decrypts the corresponding committed ballot *x* using *k* and retrieves the ballot *b*, counts the votes, and announces the voting results.

The Sensus protocol, for a large part, is the same as Fujioka et al.'s protocol. It assumes that all communication between voter and election authorities occurs over an anonymous channel. What is different in Sensus is that it uses one more central facility called *pollster* and that the tallier does not wait till the end to process votes. The latter is done by modifying the opening and counting phases. In particular, after the collecting phase, the tallier signs the encrypted ballot *x* and returns it to the voters as a receipt. Upon receiving the receipt, the voter sends the tallier the ballot decryption key *k*, and the tallier uses the key to decrypt *x* to obtain *b* and add the vote to the tally. Sensus still relies on voters to perform verification as in the opening phase of Fujioka et al.'s protocol. The pollster acts as a voter's agent, performing all cryptographic and data transfer functions on a voter's behalf.

Next, we show that using blind signatures as in these two protocols would allow the tallier to cheat the election without been detected. We note that in the preparation phase, if several voters would choose the same random keys *k* and vote in the same manner, then their encrypted ballots *x* will be exactly the same, and so they will obtain the same *y* with the validator's signature. The tallier can then replace a few (not all) of these pairs (*x*, *y*) with some other legitimate pairs (*x*', *y*'). When each of the affected voters checks for its vote, he/she will see (*x*, *y*) in the published list and hence will not detect anything wrong. To make matters worse, the tallier may generate new votes to replace duplicated votes. Since voters would use the same pseudo-random number generator provided by the protocol to generate secret keys *k*, and since in a large-scale election many of the votes will be the same, it is likely that many of the pairs (*x*, *y*) will be the same. This would make the attack successful, which would violate the accuracy criterion. While it is theoretically possible to make all the random keys distinct, in practice this would not be easy to guarantee because keys are generated by individual voters.

Fujioka et al. [F+93] noted that the validator could submit votes for voters who decide to abstain. They then suggested that voters who abstain should submit a blank ballot to avoid this from happening. This is hardly a practical solution because if the voters decide to abstain, they probably would not take the time to submit blank ballots either. Likewise, the voters who abstain cannot be relied upon to make sure that no votes were cast for them. To solve this problem, it may be possible to have some sort of time expiration on the ballots. This however, may generate more problems.

Another drawback with the Sensus protocol and Fujioka et al.'s protocol is that they rely on anonymous communication channels to provide anonymity. But anonymity is hard to guarantee over the Internet. Although there are services that offer the ability to browse the Web anonymously, such as anonymizer.com, the only way to guarantee that all voters use these services is to force them to use certain sites. However, voters cannot know, with any certainty, that these sites do not collaborate with any of the central facilities involved. Cranon and Cytron [CC97] suggest that an anonymous channel could be secured through the use of a chain of World Wide Web facilities. The problem with this solution is that some organization must configure this to occur. It would be difficult to ensure the voters that none of the Web facilities in the chain are secretly collaborating with the authority. The task of anonymity on the Web may have been made even more complicated with the recent introduction of embedding CPU identification numbers into Intel's Pentium III chips. These numbers can be

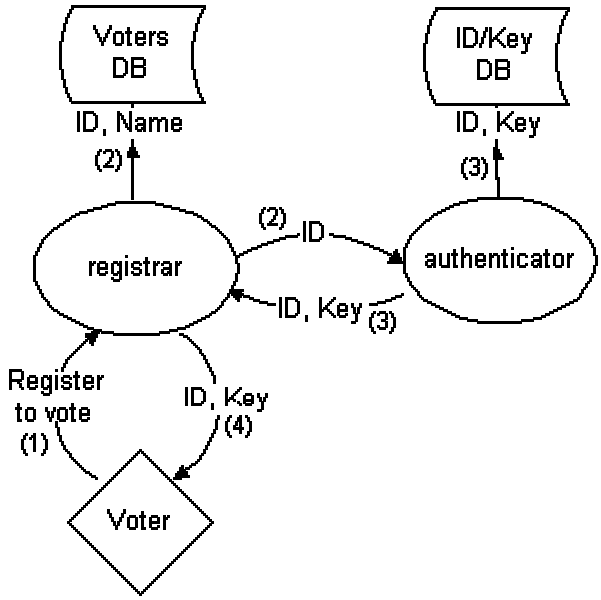
broadcast over the Internet, identifying the voter's Internet connection and the machine from which they are casting their votes. This would violate the privacy criterion.

Finally, in these two protocols, voters are relied upon to verify that their votes were counted. This is not practical, especially for voters who do not have convenient Internet access. These voters would have to revisit a polling place to verify their votes after the voting results are announced. Therefore, Sensus violates the simplicity and the verifiability criteria.

4. The proposed protocol

Our protocol does not use blind signatures or require anonymous communication channels. Instead, our protocol uses a secure form of communication (e.g. HTTPS in Netscape) for all transactions. Our protocol uses six central facilities. They are the *registrar*, the *authenticator*, the *distributor*, the *counter*, the *matcher*, and the *verifier*. The responsibilities of these facilities will be explained below when we detail our protocol. Our protocol consists of only four phases (procedures).

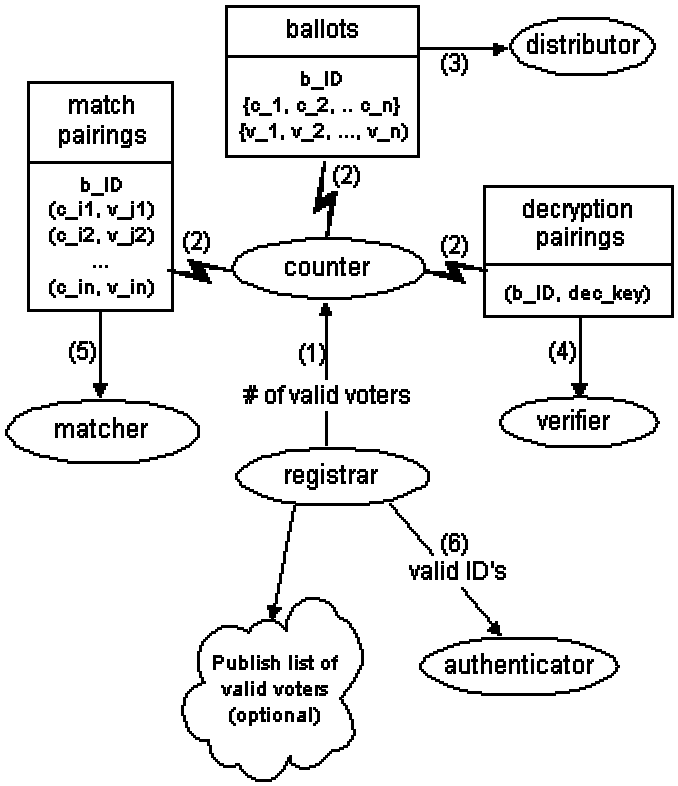
Registration phase.



1. In order to vote, a voter must first register with the registrar to identify himself as an eligible voter.
2. Upon registering, the registrar assigns a unique identification number to the voter, places the voter's name and ID in the registered voter list, and sends the ID without the name to the authenticator.
3. The authenticator generates a unique pair of public/private keys for the ID it received, stores them in a list, and sends the pair of the public key *s* and the ID to the registrar.
4. The registrar then sends the pair back to the voter. (In so doing, the authenticator will not know whom the given key *s* belongs to without conspiring with the registrar.)

Remark. The key *s* may be valid for a long time for multiple elections, or could expire after a given time. If the key were to be kept for a long duration, it would probably be best to have the voter encrypt it with a password of his/her choice, so that no one else could use it. The original, unencrypted key would be destroyed and the encrypted key (still denoted by *s*) would be stored instead. This would also allow a voter's district to store the voter-encrypted keys to prevent the key from being lost or damaged, without worrying about someone getting access to the key. Voters just need to retrieve the key from their district, or floppy disk, before voting.

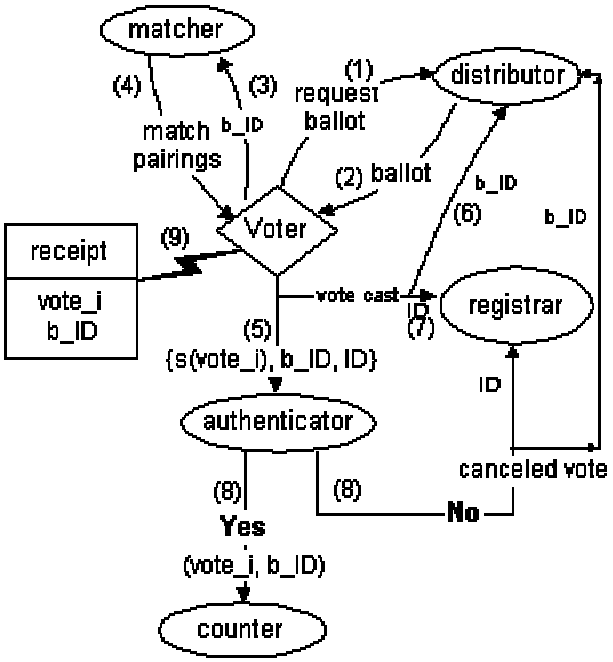
Pre-voting phase.



1. The registrar sends the number of eligible registered voters to the counter.
2. The counter generates a larger number of ballots than the number of registered voters. Each ballot consists of three things: each of the choices on the ballot, an encrypted version of each choice, and a ballot ID. The counter keeps record of the decryption key and the ballot ID for each ballot so that the counter can later decrypt the cast votes.
3. The counter sends the ballots to the distributor.

4. The counter sends a copy of the decryption table to the verifier.
5. The counter sends the match pairings (pairs of a ballots encrypted and decrypted choices) to the matcher.
6. The registrar sends the authenticator a list of ID's that are eligible for the given election. If desired, the registrar may publish the names of these voters.
7. If desired, the verifier can check the ballots and pairings to confirm that they were properly generated.

Voting phase.



1. When the voter wishes to participate in the election, he/she contacts the distributor and asks for a ballot.
2. The distributor randomly selects a ballot and sends it to the voter.
3. The voter's web browser requests the matching pair for the received ballot from the matcher.
4. The matcher sends the voter the appropriate matching pair.
5. The voter then signs the encrypted version of the desired vote using his/her signature key *s* and sends them to the authenticator, along with the ballot's ID number, and the voter's own ID.
6. The voter's Web browser informs the distributor that the ballot with the given ballot ID has been cast. (In so doing, the distributor has a record of how many votes are actually cast, and by which ballots. This will prevent any facility from generating votes for unused ballots, solving a major problem in many of the previously discussed protocols.)
7. The voter's Web browser informs the registrar that the voter has cast a vote, but it is not required to tell the registrar which ballot ID it used.
8. The authenticator first checks the signature to authenticate the voter. The authenticator then verifies that the authenticated voter is permitted to vote in the given election. Once authenticated, the authenticator passes only the legitimate encrypted vote and the ballot's ID to the counter. If authentication fails, the authenticator will notify the voter that he/she is not allowed to vote. The authenticator would then notify the registrar and distributor with a cancellation.
9. The voter's browser generates a receipt when the authenticator confirms receiving the ballot packets.

Announcement phase.

The announcement phase requires no interaction between the different facilities. Each facility merely releases certain information to the public. To verify the integrity of the election, the verifier facility compares certain published lists. An individual voter could also compare some of these lists. The integrity of the election does not require a voter to do so, but allowing a voter to perform such checks increases the security as explained in Lemma 3 of Section 6.

- The counter decrypts the votes it has received and tallies the vote.
- The authenticator publishes list #1 containing the encrypted ballots and the ballot ID.
- The counter publishes list #2 containing its version of list #1. Both lists 1 and 2 should be identical.
- The authenticator publishes list #3 consisting of all voter IDs that cast ballots (in numerical order).
- The registrar looks at list #3 and confirms that only valid voters voted. (This list could also be published if desired.)
- The verifier confirms that lists 1 and 2 are identical. (To prevent cover-ups, it may be desirable to have lists 1 and 2 be sent to the verifier before they are published.)
- The verifier uses list 1 and the decryption table (from counter in the pre-voting phase) to confirm the results published by the counter.
- Voters can look at lists 1 and 2 to see their votes on both of these lists.
- The distributor looks at lists 1 and 2 to be ensured that only legitimate ballots appear. Any illegal ballots can than be removed and the results recalculated. The distributor could also release its list of ballot ID's, but this should be done after the authenticator and the counter released their encrypted ballot lists.
- The counter announces the election results, which can be verified by the verifier.

Remark. Revealing the source code, much in the same

way as with PGP, could allow laymen to check the validity and honesty of the facilities.

5. Security measures and implementation

To ensure that elections are held fairly, we must develop security measures to prevent individual modules of our voting protocol from conspiring with each other. We require that each of the facilities generate a pair of public and private keys of its own. These pairs should be replaced from time to time. To keep elections from being delayed, we recommend changing the keys between elections. We assume that not all of the facilities can be compromised at the same time. This is a reasonable assumption, for there is little one can do if all of the facilities are compromised simultaneously. In any conventional voting system, the overall security and integrity rely on humans. This means that the integrity of a traditional election is only as strong as that of the people running it. We will use a public-key encryption/decryption scheme where keys commune. To prevent facilities from communicating illegally, all facilities will monitor the facility-facility communication channel.

5.1. Data protection

Each facility is required to encrypt its database (list of data) on the fly, *e.g.*, one record at a time, using the public keys of all the facilities. By doing so, the only way to completely decode a piece of data would be to acquire the secret keys of all severs, which, by our assumption, is impossible. Because the database is encrypted piece by piece, the facility can easily extract the portion of the data from the database it needs to see and then sends it to the other facilities to decrypt it.

It is not necessary to encrypt election results, as they will be released at the end of the election. It would also be very easy to see any discrepancy in the results when all of the lists are released. It is necessary to encrypt the database of the distributor to protect the ballots that have not been given out.

5.2. Security of Communication Channels

We have two type of communication to deal with. The first type is between facilities, and the second type is between a voter and a facility.

Facility-facility communication. For this type of communication, we need to ensure that these communications cannot be intercepted or altered; we also need to ensure that facilities do not collaborate to compromise the integrity or anonymity of the election. We accomplish both of these goals using the following protocol. When facility A wants to transmit data to facility B, facility A sends the encrypted data to a randomly selected third facility C. Facility C then decrypts the data with its own secret key, verifies that the size and the structure of the data it received have not been altered, and sends the data to another randomly selected facility D. The process is continued until the data finally reaches facility B, and facility B will be able to read the data after it uses its private key to decrypt the data.

Since intermediate facilities cannot completely decrypt the data, they will not know what exactly is being sent. The protocol can ensure that the information being sent is of legitimate size and structure. The only way for an intermediate facility to cheat would be to rearrange the information so it matches this size and structure. This would cause some information, such as some of the ballots to be left off, but the other facilities would be able to notice this when tabulation occurs.

Since facilities could manipulate this process by breaking the illegal data into small parts and reporting sizes that make the data appear legitimate. The facilities should each keep a log of the status of the protocol. This way communication can only occur between two facilities at appropriate times and should be limited as to how many communications they are permitted.

To reduce the amount of traffic, as well as decryption computation, communication between facilities should be done in large blocks. For instance, the counter should send all of the ballots to the distributor, and the authenticator should send the counter encrypted ballots in a large number of blocks.

Voter-facility communication. Since we are dealing with the Internet, the most logical form of security for the interaction between the voter and the central facility would be to use HTTPS. HTTPS is already considered to be a secure form of communication for the Internet. It is considered to be a *de facto* standard, and as long as it is viewed as such, it would be reasonable to use HTTPS. If circumstances cause a new standard to arise, this new standard should be adopted for this type of communication.

The only alteration to the HTTPS protocol we will have to deal with is the fact that when the voter is being sent something it would be encrypted. Of course, the facility also would not be able to look up the requested information. Therefore, the facility encrypts the database and sends it to the other facilities to remove their encryption. The facility gets the information back, decrypts it with its secret key and then looks up the requested information.

6. Proof of anonymity and security

In this section provide proofs that our protocol satisfies all nine of the criteria defined in Section 2. Recall that we assume that not all facilities collaborate at the same time. We first prove the following lemma.

Lemma 1. If no facility knows all other facilities' secret keys, then any collaboration among facilities can be detected by a non-collaborating facility.

Proof. We note that each facility's data is stored in an encrypted form with all the other facilities' public keys. The collaborating facilities cannot bypass the other facilities, because without them the data cannot be decrypted. Hence, the only way for two facilities A and B to collaborate is to cheat: The sending facility A does not encrypt the data and sends the data directly to the receiving facility B. Such activities can be detected by a non-collaborating facility C by monitoring the data transactions in the follow ways.

Case 1. Facility A specifies that facility B is the destination facility and sends the data directly to B. Then the non-collaborating facility C can find out that A cheats because C must receive the data before B does.

Case 2. Facility A specifies that facility B is not the destination, but picks B to be the first facility to pass the data. Then the non-collaborating facility C can find out that A cheats after a few rounds of transactions because A is supposed to randomly pick a third facility to send the data and C should have a chance to receive it in a few rounds.

The similar proof can be applied for the case where more than two facilities collaborate. This completes the proof.

Based on Lemma 1, we assume that no facilities collaborate in the rest of the proofs presented below.

Lemma 2. The democracy criterion is satisfied.

Proof. We assume that no cheating occurs in the registration phase; otherwise, there is little we can do no matter what voting protocol is used.

We first show that only eligible voters are allowed to vote. If an ineligible voter tries to vote, the authenticator can notice this and will not allow the vote to be cast. If the authenticator cheats by allowing an ineligible voter to participate in the election, the registrar will notice this when it receives the list of ID's that voted. If the registrar allows an ineligible voter to vote, then either too many voters would be permitted to vote, or an eligible voter would be denied the right to vote by the authenticator. In the first case, since we know the exact number of eligible voters for the given election in the registration phase, the authenticator or the counter would notice that too many people were being allowed to participate. In the second case, the voter will be notified and so the voter can challenge the registrar or the authenticator. The voter could request the registrar to inform the authenticator that he/she is eligible, which may then result in the first case.

Next, we show that each eligible voter can only vote once. If a voter tries to vote twice, the authenticator would notice that the signature key *s* and ID had already been used. Depending upon the voting scenario, the new vote would either overwrite the old vote, or it would simply be ignored. If the authenticator tries to pass the new vote on anyway, it would have to place it in place of someone else's vote, because otherwise the lists posted at the end would not match in length. The registrar, however, has its own list of voters, and their ID's that actually voted. Eventually, there would be a conflict with these lists. This completes the proof.

Lemma 3. The accuracy criterion is satisfied.

Proof. Due to the fact that voters are given a receipt, and that they are allowed to view the published lists as described in the Announcement Phase, a voter's vote cannot be altered, duplicated, or removed without being detected. An attempt to alter or remove votes would be futile since the cheating party would not know which voters are going to check for their ballot. If a cheater changes a ballot and the voter whom cast the ballot examines the list, it would be evident that fraud had occurred. Appropriate measures could than be taken to remedy the error. In a large scale election, the cheater would be required to alter many ballots, increasing the likely hood of being caught.

There are three kinds of votes that are considered invalid, namely, votes made by ineligible voters, votes made by eligible voters but the votes are in incorrect formats, and votes generated by central facilities for unused ballots. For the first kind of invalid votes, as shown in the proof of Lemma 2, they will be detected before the final result is announced, and so they will not be counted. For the second kind of invalid votes, the counter will not be able to tally them since they are in wrong formats. For the third kind of invalid votes, since many lists are published at the end of the election, no facility can generate votes for unused ballots without being detected. This completes the proof.

Lemma 4. The privacy criterion is satisfied.

Proof. The only facility that can see the voters' names is the registrar. The registrar, however, can only see the encrypted ballot cast by a particular voter's ID. The registrar has no way to decrypt this vote without collaborating with the counter. We have shown in Lemma 1 that this cannot occur.

Lemma 5. The verifiability criterion is satisfied.

Proof. Voters can be sure that their votes were tabulated by verifying that their ID and encrypted key are in the lists posted by the authenticator and the counter. Moreover, the voters are not relied upon to verify their votes because this is the job of the verifier. Although we do not require voters to check their ballots, it can be assumed that some will. Therefore, since the verifier does not know who will check their ballots, the verifier cannot cheat without being detected.

Lemma 6. The simplicity criterion is satisfied.

Proof. The voter is required to do very little, except that he/she needs to register and vote. The facilities do the majority of the work, with the voter's computer doing

very minor calculations, and voters can vote with minimal equipment and skill.

Lemma 7. The mobility criterion is satisfied.

Proof. This is straightforward since our protocol is to be used over the World Wide Web.

Lemma 8. The efficiency criterion is satisfied.

Proof. As we mentioned earlier that in our protocol, the facilities do the most of the computations. In particular, all the calculations, except the signatures, are done before the voting even occurs. This means that very little time is consumed in the actual voting process. The main delay in voting would be the actual network communication. If the voting population were divided into districts the network delay would be minimal. Keeping the facilities in a close physical proximity, connected via a high-speed network, would also minimize delays. We can run the facilities using powerful computers (or special-purpose computers) to increase efficiency.

Lemma 9. The scalability criterion is satisfied.

Proof. Since our protocol is to be run over the World Wide Web, it is easily scalable and divisible. If districts are desired or needed, our protocol will compensate for that by having each district running its own facilities. Large-scale elections would run smoother if they were partitioned, but it is not necessary to do so.

Lemma 10. The responsibility criterion can be satisfied.

Proof. As we mentioned before that the responsibility criterion is an optional requirement, which is not required in the US elections. But it is desirable in Australian elections. If this criterion is desired, the registrar can easily make it possible by publishing the names that have voted.

7. Additional Properties

In addition to the properties we proved in Section 6, we outline below some additional properties of our voting protocol.

- Our protocol can be easily modified to allow the facilities to hold multiple elections simultaneously. For instance, we can participate in a nationwide election at the same time we vote for local officials or ordinances. This could be achieved by adding an election ID to the ballots. The ID would tell the facilities what election the given ballot is for. Voters would request a set of ballots instead of a single ballot.
- Voters may be allowed to change their vote. This could be done in one of two ways. First, authenticator holds all votes till the end, to change a vote, the user just resubmits their vote. The authenticator throws out the old vote and keeps the new one. Second, when the authenticator sees that the voter has already cast his/her ballot for the given election, the authenticator asks the counter to remove the ballot from its list. The authenticator then sends the new vote to the counter. As an added benefit of this property, we can make vote selling more difficult, because the buyer now has to lock the seller until the end of the election to prevent the buyer from changing his/her vote.
- Our protocol can handle many types of elections (*e.g.*, several candidates, picking multiple candidates, write-in), with very limited modification.
- Interested parties could have their own facilities designed to check the integrity of the election.
- Using the distributor facility, we are allowing elections to occur on the Internet without worrying about hiding or masking IP addresses. The distributor facility also provides additional reliability on the integrity of the election.

Final remark. If the parties running the individual facilities would not collaborate (e.g., due to conflict interests) and they are in a secure environment, then some of the security measures such as encrypting data using public keys of all facilities could be removed.

Bibliography

[Be87] J. Benaloh. *Verifiable Secret-Ballot Elections*. PhD. Thesis, Yale University, 1987.

[BT94] J. Benaloh and D. Tuinstra. Receipt-free secret-ballot elections. In *Proceedings of the 26th ACM Symposium on Theory of Computing*, pages 544-553, ACM Press, 1994.

[Ch82] D. Chaum. Blind signatures for untraceable payments. In *Blind Signatures for Untraceable Payments*, D. Chaum, R. Rivest, and A. Sherman, eds., pages 199-203, Plenum Press, 1982.

[Ch81] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communication of the ACM*, 24(1981), pp. 84-88.

[Ch88a] D. Chaum. The dinning cryptographers problem: unconditional sender and recipient untraceability. *Journal of Cryptography*, 1(1988), pp. 65-75.

[Ch88b] D. Chaum. Elections with unconditionally secret ballots and disruption equivalent to breaking RSA. In *Proceedings of Advances in Cryptology (EUROCRYPT'88)*, vol. 330 of *Lecture Notes in Computer Science*, pages 177-182, Springer-Verlag, 1988.

[Co86] J.Cohen. Improving privacy in cryptographic elections. Yale University Tech. Rep. DCS/TR-454, 1986.

[CF85] J. Cohen and M. Fisher. A robust and verifiable cryptographically secure election scheme. In *Proceedings of the 26th IEEE Annual Symposium on Foundations of Computer Science*, pages 372-382, IEEE Computer Society Press, 1985.

[C+96] R. Cramer, M. Frankin, B. Schoenmakers, and M. Yung. Multi-authority secret ballot elections with linear work. In *Proceedings of Advances in Cryptology (EUROCRYPT'96)*, vol. 1070 of *Lecture Notes in*

Computer Science, pages 72-83, Springer-Verlag, 1996.

[CGS] R. Cramer, R. Gennaro, and B. Schoenmakers. A secure and optimally efficient multi-authority election scheme. Manuscript acquired from rosario@watson.ibm.com.

[Cr96] L. Cranor. Electronic voting: computerized polls may save money, protect privacy. *ACM Crossroads* (Electronic Journal), 1996.

[CC97] L. Cranor and R. Cytron. Sensus: A security-conscious electronic polling system for the Internet. In *Proceedings of the Hawaii International Conference on System Sciences*. Wailea, Hawaii, 1997.

[F+93] A. Fujioka, T. Okamoto. and K. Ohta. A practical secret voting scheme for large-scale elections. In *Proceedings of Advances in Cryptology (AUSCRYPT'92)*, vol. 718 of *Lecture Notes in Computer Science*, pages 244-251, Springer-Verlag, 1993.

[Iv91] K. Iversen. A cryptographic scheme for computerized general elections. In *Proceedings of Advances in Cryptography (CRYPTO'91)*, vol. 576 of *Lecture Notes in Computer Science*, pages 405-419, Springer-Verlag, 1991.

[MV98] Y. Mu and V. Varadharajan. Anonymous secure e-voting over a network. In *Proceedings of the 14th Annual Computer Security Application Conference*, pages 293-299, IEEE Computer Society. 1998.

[Na90] M. Naor. Bit commitment using pseudo-randomness. *In Proceedings of Advances in Cryptology (CRYPTO'90)*, vol. 435 of *Lecture Notes in Computer Science*, pages 218-229, Springer-Verlag, 1990.

[Ne93] P. Neumann. Security criteria for electronic voting. In *Proceedings of the 16th National Computer Security Conference*, pages 478-481, 1991.

[NS91] H. Nurmi and A. Salomaa. A cryptographic approach to the secret ballot. *Behavioral Science.* 36(1991), pp. 34-40.

[NS] H. Nurmi and A. Salomaa. Secret ballot elections and public-key cryptosystems. Manuscript.

[N+91] H. Nurmi, A. Salomaa, and L. Santean. Secret ballot elections in computer networks. *Computers & Security*, 10(1991), pp. 553-560.

[Pf84] A. Pfitzmann. A switched/broadcase ISDN to decrease user obervability. In *Proceedings of the International Zurich Seminar on Digital Communication*, pages 183-190, IEEE Computer Society Press, 1984.

[SK94] K. Sako and J. Kilian. Secure voting using partially compatible homomorphisms. In *Proceedings of Advances in Cryptology (CRYPTO'94)*, vol. 839 of *Lecture Notes in Computer Science*, pages 411-424, Springer-Verlag. 1994.

[Sal96] A. Salomaa. *Public-Key Cryptography*. 2nd edition. Springer-Verlag, Berlin, 1996.

[Sch96] B. Schneier. *Applied Cryptology*, 2nd edition. John Wiley & Sons, New York, 1996.