E-Voting Report
CS 386
May 14, 2009

Cabrera, Guillermo
Gopinath, Divya
Kulkarni, Nupur

1. Introduction:

Electronic voting or e-voting refers to the use of electronic technology for creating votes, use of electronic medium such as the internet or telephones for transmission of ballots and electronic tallying or counting of votes. Traditionally voting is conducted by setting up voting stations where voters need to be present physically on the day of election. The identity and eligibility of voters are verified manually by government representatives. Voters cast their vote by marking on paper ballots inside a secure voting booth. There are significant problems with the conventional voting process as catastrophically proven in the 2000 Presidential Election. Some of them are listed below.

- Administration Difficulties:
  Printing of paper ballots or punched cards in large numbers, securely transporting them, counting them manually are highly labor intensive and time consuming. In addition, paper ballots are vulnerable to being corrupted and misused.
- User Interface Problems:
  Paper ballot schemes are not very convenient to use and often suffer from problems due to under-voting or over-voting. Also, in some cases such as the "butterfly ballots" the ballot layout is so confusing that voters mistakenly cast votes for individuals they don't intend voting for.
- Accessibility:
  Voting systems which involve physical presence of voters at the voting stations may not be easily accessible to people with disabilities or special needs.

Electronic voting can provide solutions to these issues. Use of electronic medium for transmission and counting of ballots can greatly speed up the process and ease administration of elections. Computers and cell-phones could be designed to provide more convenient and customizable user interfaces. E-voting helps every eligible voter to participate in the elections without any location constraints. Thus electronic voting displays the potential to be an efficient and convenient solution to conduct elections and increasing voter participation. However, electronic voting process faces many challenges such as those mentioned below,

- Authentication of Voters:
  Since the e-voting process is accessed remotely by voters, ensuring that only eligible voters are allowed to participate in the elections is a challenge. In addition, impersonation or misuse of registered voter IDs also needs to be prevented. Also, voters should not be allowed to cast vote more than once.
- Verifiability of correctness of voting process:
  To have confidence in the election results, people must believe that the election tasks have been performed properly without fraud. There must be a universally acceptable way to verify the correctness of the voting process.
- Voter and Data Anonymity:
  The prevention of election fraud is made more difficult by the requirement that votes remain private. Observers may not observe a ballot until after it has been placed in a ballot box, and audit trails must not provide the ability to link a ballot back to the voter who cast it.

In order to determine whether an e-voting system is sound, reliable and secure, it is useful to develop a set of criteria for evaluating system performance. The following is a set of desirable characteristics for electronic polling systems which incorporates the characteristics of most systems described in the electronic voting literature [2], [10].

- **Accuracy:** A system is said to be accurate if (1) it is not possible for a validated vote to be eliminated from the final tally, (2) it is not possible for an invalid vote to be counted in the final tally and (3) it is not possible for an altered or tampered vote to be included in the final tally. Fully accurate systems ensure that the final vote tally is perfect, either because no inaccuracies can be introduced or because all inaccuracies introduced can be detected and corrected. Partially accurate systems can detect but not necessarily correct inaccuracies.
- **Invulnerability:** A system is said to be invulnerable if (1) it permits only eligible voters to vote and (2) it ensures that each eligible voter can vote only once.
- **Privacy:** A system is said to be private if (1) neither election authorities nor anyone else can link any ballot to the voter who cast it and (2) no voter can prove that he or she voted in a particular way to prevent vote buying and extortion.
- **Fairness:** Nobody can guess the votes, trend of voting or intermediate results before the final counting.

- **Verifiability:** A system is said to be *Universally Verifiable* if anyone can independently verify that all valid votes have been counted correctly. Some e-voting systems are verifiable by voters, who can verify their own votes and correct mistakes without sacrificing privacy. Less verifiable systems might allow mistakes to be pointed out but not corrected.
- **Receipt-Freeness:** A system is said to be receipt-free if it prevents a voter from proving to others that s/he voted for a particular candidate. Such a feature is essential in order to prevent the use of coercion to force voters to vote in a particular way (vote buying and selling).
- **Convenience:** A system is said to be convenient if it allows voters to cast their votes quickly, in one session, and with minimal equipment or special skills.
- **Flexibility:** A system is said to be flexible if it allows a variety of ballot question formats and can support multiple candidates by not being restricted to single bit yes/no votes.
- **Robustness:** Achieved if a system can have a successful election process, in the presence of problems with some dishonest authorities or servers.

The approaches proposed over the years for constructing e-voting schemes could be divided into three main categories.

- **Blind Signatures:** These schemes use the property of blind signatures [3] to enable verifiability of the correctness of the voting process without compromising on voter privacy.
- **Homomorphic Encryption:** These schemes use the homomorphic property of ElGamal Encryption schemes to enable verifiability of the correctness of the voting scheme publicly. They also aim to satisfy the receipt-free property of e-voting schemes in order to prevent coercion.
- **Mix-Nets:** Mix-net is a network of servers (mix-center) which takes a set of cipher texts as input, and outputs the corresponding plaintexts according to a secret permutation. These have been utilized in e-voting schemes to conduct fair elections maintaining voter privacy and anonymity.

In section 2 of this report, an e-voting scheme based on blind signatures would be presented. This is a simple and efficient method for conducting large scale elections and satisfies most of the basic criteria required to ensure secure e-voting. However it suffers from drawbacks such as not being receipt-free and universally verifiable. The second paper, presented in section 3, discusses a scheme based on homomorphic encryption. In addition to meeting the basic properties of e-voting, this scheme is also universally verifiable and receipt-free. Section 4 presents the discussion of a paper which explores the use of mix-nets in e-voting. A robust and receipt-free protocol is proposed in this paper. Section 5 compares the strengths and weaknesses of the three schemes and discusses the recent developments in the field of e-voting.

2. E-Voting Scheme Using Blind Signatures. (*Report by Divya Gopinath*)

This section presents a discussion of an electronic voting scheme proposed in the paper "**A Practical Secret Voting Scheme for Large Scale Elections" by Atsushi Fujioka, Tatsuaki Okamoto, Kazuo Ohta in 1992 [1].** The explanation of the protocol is structured as follows; Section 2.1 covers related work in e-voting which motivated the proposal of this protocol, Section 2.2 gives an overview of blind signatures and bit-commitment schemes which are used in this protocol, Section 2.3 gives a outline of the protocol (participants , assumptions and overview of the phases involved), Section 2.4 explains the protocol with the cryptographic details, Section 2.5 explains the e-voting properties satisfied and Section 2.6 highlights the strengths and drawbacks of the protocol.

2.1 Motivation

The protocols that had been proposed in the field of e-voting before 1992 could be mainly divided into two categories based on the issues they addressed. One set of schemes focused on allowing only eligible voters to vote in a simple and scalable fashion. They employed one or two agencies such as a validator/administrator and tallier/counter [8], [9] which performed the duties of authenticating voter eligibility and processing votes to declare election results respectively. But these schemes were vulnerable to loss of fairness and voter privacy due to collusion or fraud by the agencies. On the other hand, there were schemes proposed which either involved only voters managing all procedures of the election [6], or ballots being sent in highly encrypted form to distributed centers to maintain voter privacy. These protocols suffered from being computationally complex. The motivation of this paper was to address both these issues by proposing a scheme which was practical and scalable without compromising on voter privacy and fairness of the elections.

2.2 Overview of Blind Signatures and Bit-commitment

- **Blind Signatures:**
  David Chaum first introduced the concept of blind signatures in early 1980's [3], [5] where he suggested that they could be used for secret ballot elections. Blind signatures are digital signature schemes used in scenarios wherein the message author and signer are different parties. Message author needs to obtain signature from the signer, hiding the actual contents of message from the signer. A physical analogy is putting the letter (message) to be signed into a sealed envelope along with carbon lining paper and sending it to an authority. Authority signs on the outside of the envelope without being able to read the contents of the letter. When the letter is removed from the envelope by the message author, the signature is imprinted on it due to the carbon lining.
  The cryptographic implementation of this concept would be as follows; Authority has a signing function or private key Ra. Message author has a private function b and its inverse b' such that the following properties hold good [3],[4].
  Blinding: m can't be guessed or obtained from **b (m).**
  Commuting: There exists an inverse **b' such that b' (Ra<b (m)>) = Ra<m>**
  Message author sends b (m) to authority. Authority signs it as Ra<b (m)> and sends the signature back. Author applies b' (Ra<b (m)>) to retrieve admin's signature on the original message, Ra<m>.
- **Bit-Commitment Scheme:**
  Bit-commitment schemes [7] are used to address scenarios wherein a party A needs to send a message to party B, such that the following conditions hold,
  Concealing: B can't obtain or decipher message until A wishes to reveal it.
  Binding: Once message is sent to B, A has to commit to the message and can't change the message in the reveal stage.
  A physical analogy would be that A puts the message into a box and locks it. A sends the box to B in a commit stage. A then sends the key to unlock the box in a subsequent reveal stage. The cryptographic implementation of this concept would be; A generates a private /secret key pair **(se, sd).** A sends message m encrypted with se, ***m'= se<m>*** in the commit stage. B doesn't know sd and hence can't decipher m from m'. In a subsequent reveal stage, A then sends sd to B using which it deciphers m.
  sd <m'> = sd<se<m>> = m.

2.3 Outline of the protocol

The e-voting scheme proposed in this paper involves the following participants,
Voter: An individual who has been verified to be eligible to vote (in a registration phase conducted prior to the start of this protocol).
Administrator: An authority who checks if a voter who wants to cast a ballot in the election is registered or not.
Counter: An agency/individual responsible for collecting ballots from voters during election, counting the votes and declaring the winner.
The model of the protocol makes the following assumptions about the system under consideration.

- Anonymous channel exists between voters and counter, hence the messages travelling from voters to counter can't be associated /linked to the voter sending it.
- Messages from voters will not arrive at the admin end and the counter end in the same order.

Presented below is a basic overview of the steps involved in the protocol. There are 6 phases in this scheme – **Preparation, Administration, Voting, Collection, Opening and Counting.**

- **Preparation Phase:**
  The voter prepares the vote and encrypts it with a secret key (commit phase of a bit-commitment scheme). S/He then blinds this committed ballot using a blinding technique. S/He then signs the blinded ballot and sends it to the administrator.
- **Administration Phase:**
  The administrator checks the eligibility of the voter to vote and verifies the signature of the voter. He also ensures that the voter had not already applied for validation. If these checks pass, s/he then signs the blinded ballot with the admin signature chosen for the election and sends it back to the voter. The admin then publishes a list containing the voter's ID and blinded ballots which were given signatures.
- **Voting Phase:**

The voter applies the inverse of the blinding technique on the signature of the blinded ballot. This retrieves the original un-blinded ballot with the admin's signature on it, which acts as the admin's certificate for the voter's eligibility. S/He then sends the ballot (encrypted version of the actual vote) and the certificate to the counter via an anonymous channel.

- **Collection Phase:**
  The counter checks that the ballot has the admin's signature. If the ballot is valid, it is added to a list which is published after the time allocated for the collection phase is over. This published list of ballots is open for verification by voters.
- **Opening Phase:**
  Voter sends to the counter the secret decryption key to reveal the vote. The counter opens the vote and publishes it on the ballots list beside the respective encrypted ballot.
- **Counting Phase:**
  The counter counts / tallies the votes and declares the winner.

2.4 Detailed explanation of the protocol (with cryptographic details)

Lorrie F. Cranor and Ron Cytron presented a system named "Sensus" in 1997[2], which is an implementation of the protocol scheme presented in this paper. Their work gives concrete details on how certain cryptographic functions are implemented. I have included details of the "Sensus" system as well in this discussion.

Notations:

- **vi** – Vote cast by voter.
- **IDi** – Voter's registration id.
- **Bi, Ri** – Public and Private keys of voter.
- **se, sd ,bi - se** is a secret encryption key and **sd** is a secret decryption key used by voter to apply bit-commitment scheme on vote (**vi**) to generate a ballot, **bi.** *A ballot is an encrypted version of a vote.*
- **Ba, Ra** – Public and Private keys of Admin generated for this election.
- **(IDi, Bi)** - List of registered voters and their public keys (Admin has this).
- **b (bi, ri) , bbi** – **b** is a blinding function used by voters to blind their ballots **bi** with a random number (confounder) **ri** to generate a blinded ballot (**bbi**). Nobody can guess value of bi from bbi.
- **b' is the inverse of b** such that **b'(Ra<bbi> , ri) = Ra<bi>** ,where Ra < bbi > is the admin's signature on the blinded ballot and Ra<bi> is the admin's signature on the un-blinded ballot .

Preparation Phase [figure 1]:

Voter does the following,

- Prepares vote ***vi.***
- Encrypts the vote with secret encryption key ***se*** to generate ballot ***bi***.
- Chooses a random number ***ri***. Blinds the ballot ***bi*** using ***ri*** as a confounder. This is done by applying the blinding function ***b(bi,ri)*** to generate the blinded ballot ***bbi.***
  One way to implement the blind signature function and its inverse is with the use of RSA signing scheme [3], [4].
  b (bi, ri) *is implemented as being equal to* ***bi x Ba<ri>*** ***(1)***
  Where bi is the ballot, ri is a random number, *x* is multiplication operation and Ba is the public key of the admin *(refer Appendix 7.1 for elaboration on how this is exactly implemented using the RSA encryption scheme).*
- Signs the blinded ballot using his private key ***Ri***.
- Sends his ID, blinded ballot and his signature on blinded ballot as a tuple to Admin. *In the Sensus system implemented by Cranor and Cytron, this tuple is encrypted with the public key of the Admin Ba.*

Administration Phase [figure 1]:

Administrator gets the tuple ***(IDi, bbi, s)*** from the voter, where s is the signature of the voter. Admin wouldn't be able to guess the value of the ballot (bi) from the blinded ballot (bbi) due to the presence of the random confounder ri. Admin does the following,

- It verifies the eligibility of the voter to vote by checking if ***IDi*** is present in the list of registered voters ***(IDi,Bi).***

- It obtains the public key Bi of the voter from this list and authenticates the signature of the voter by checking if
  Bi <s> = bbi.
- Checks that IDi hadn't already sent a request in the past for the same election.
- If any of the above checks fails, the administration is rejected for that IDi.
- If every check passes, admin signs the blinded ballot using his private key ***Ra*** and sends the signature ***(sbbi)*** back to the voter. This acts as a certificate provided by admin to the voter to confirm the voter's eligibility.
  sbbi = Ra<bbi> = Ra < bi x Ba<ri> > from (1)
  = Ra<bi> x RaBa<ri> = Ra<bi> x ri ***(2)*** *(refer Appendix 7.1 for elaboration on how this is exactly implemented using the RSA decryption scheme).*
- *In the Sensus system, the certificate sent back is encrypted with the public key of the voter Bi.*
- After Administration phase, admin publishes the list of IDs given valid certificates, the corresponding blinded ballots and certificates on a bulletin board viewable by everyone.

b

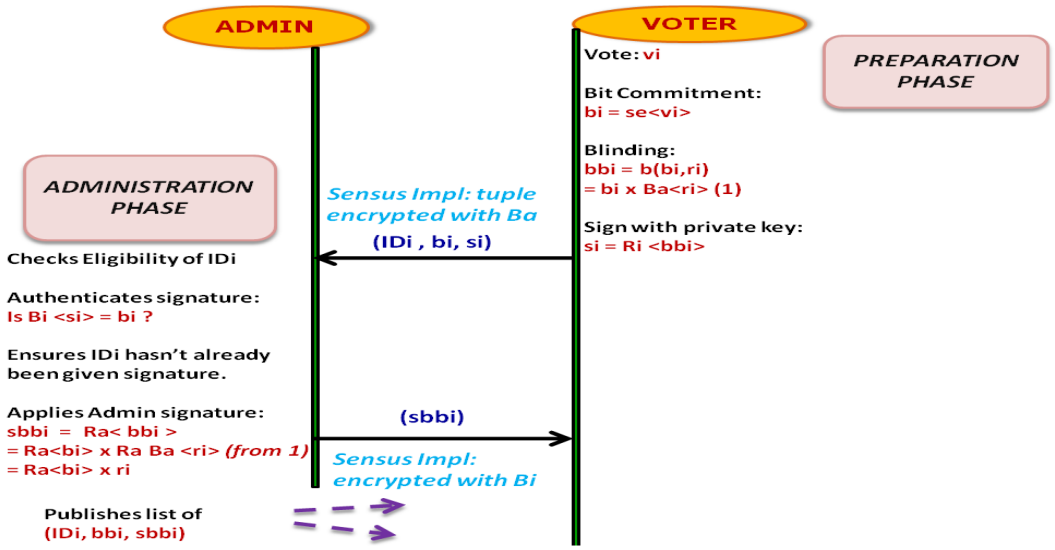

Figure 1: Preparation and Administration Phases.

Voting Phase [figure 2]:
Voter receives the signature of the blinded ballot ***sbbi.***

- Voter applies the inverse blinding function ***b'*** to retrieve admin's signature on the original ballot ***(sbi)*** from the signature of the blinded ballot ***(sbbi) .***
  sbi = b'(sbbi,ri) = b' (Ra<bbi>,ri) , where ri is the random number used in the preparation phase to blind the ballot.

  The inverse of the blind signature function b' is implemented using the RSA signing scheme as follows [3], [4].
  b' (Ra<bbi>,ri) = Ra<bbi> / ri ('/' is division operation)
  = (Ra<bi> x ri) / ri from (2)
  = Ra<bi> , signature of un-blinded ballot

- It also checks if the signature provided by admin is correct by checking if
  Ba <sbi> = bi , where Ba is the public key of admin.
- If the signature looks fine, it sends ballot and admin's certificate (signature on un-blinded ballot) to the counter via an anonymous channel.

Collection Phase [figure 2]:

Counter receives ***(bi, sbi)***, Counter does the following,

- Verifies if the certificate is valid, by applying the public key of admin (Ba) on it and checking if ***Ba <sbi> = bi***. If not, it rejects the ballot.
- If the signature is valid, it adds the ballot and the certificate to a list which is published on a bulletin board ***(index , bi, sbi or Ra<bi>),*** where ***index*** is a unique serial number on the list.
- *In the Sensus system, the counter also checks if the received ballot is* ***unique****. Also if the checks on the ballot pass, a receipt which contains the ballot and its certificate signed by the counter is sent back to the voter. This acts as a proof for considering the voter's ballot.*
- The published list of ballots is verified by voters to see if their ballot has been included. Also, the number of ballots on the list is compared with the list of IDs given valid signatures published by the admin (after Administration phase) to detect any fraud done during the voting or collection phase.

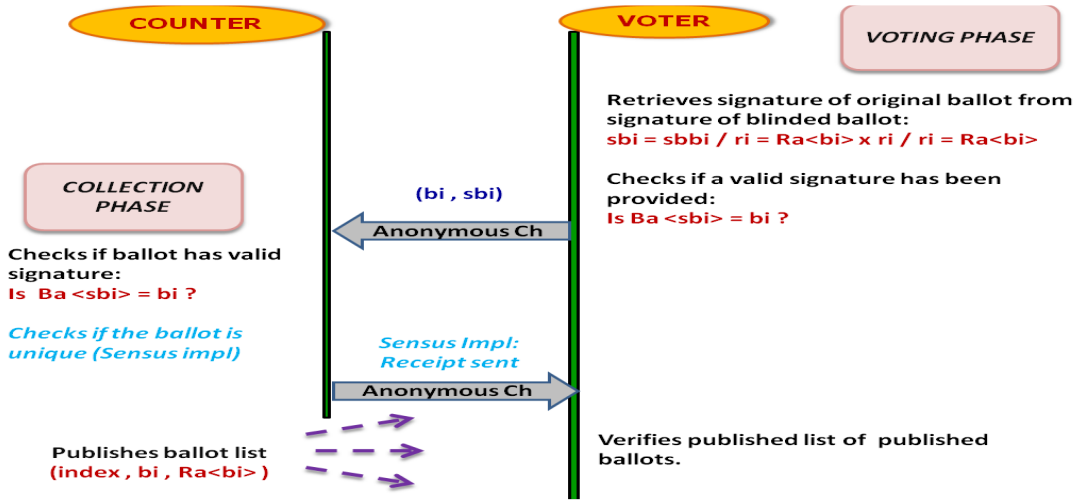

Figure 2: Voting and Collection Phases.

Opening Phase: [figure 3]

- Once the published list of ballots is verified and corrected if required, the voters send the secret decryption key ***(sd)*** to the counter. This acts as the reveal stage for the bit-commitment scheme applied on the vote.
- The counter decrypts the ballot bi to reveal the vote ***vi***, ***sd <bi> = sd se<vi> = vi.***
- The published list is then updated by placing the opened vote beside the ballot. ***(index , bi, Ra<bi> , vi)***

Counting Phase [figure 3]:

- The counter counts the votes for each candidate and declares the winner.

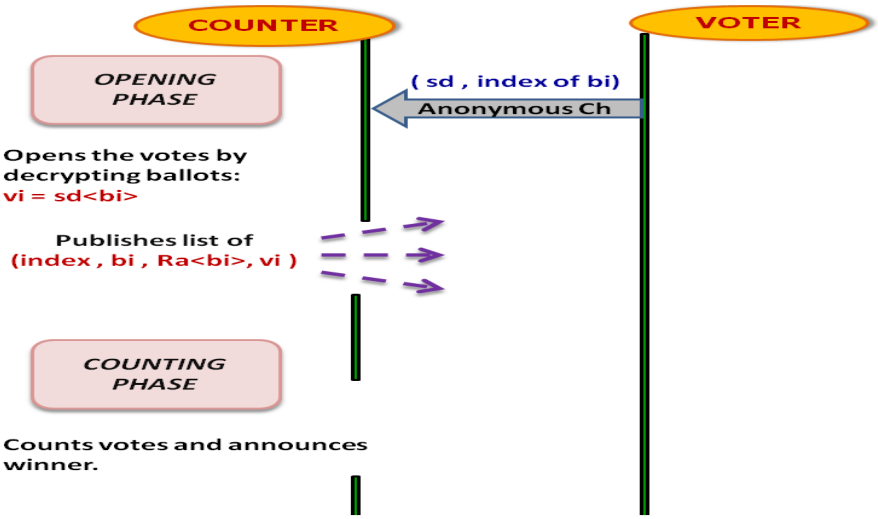

Figure 3: Opening and Counting Phases.

2.5 E-voting properties satisfied by the protocol

- **ACCURACY:** A validated vote can't be left out of the tally (referred to as **COMPLETENESS** in the main paper [1]). An invalidated vote can't be included in the tally. An altered or tampered vote can't be included in the tally.
  Attacks threatening accuracy and how they are combated.
  Accuracy could be threatened due to the counter, admin or voter doing foul play. This can be detected as will be explained in the points below. In some cases, the election result can be corrected while in others, the voting process is invalidated and restarted. Assumption: More than one party (Counter, Admin and Voter) can't collude together in doing foul play.
  - *A voter's ballot gets altered or tampered by adversary on its way to the Counter (in the voting phase) <or> Counter does foul play and ignores a valid ballot.*
    This can be caught by the voter when the list of published ballots (after collection phase) doesn't contain his ballot. He can anonymously announce his ballot and the certificate obtained from admin on his ballot to prove the validity of his ballot. *In Sensus system, the voter can show his receipt to prove that he cast a valid ballot*. Thus the election tally can be corrected.
  - *Counter doesn't open valid ballots to reveal votes or doesn't count votes properly.*
    This can be caught by voters or anybody auditing the election results by comparing the list published after opening phase (containing the ballots and votes) with the verified list of ballots published after the collection phase.
  - *A voter or adversary replays a ballot to Counter (in voting phase) <or> Counter publishes a ballot twice in the collection phase.*
    In this case, the number of ballots published after Collection phase would be more than the number of IDs given validated certificates published by the Admin after Administration Phase. To resolve this, the voters reveal the random number ***ri*** that they had used to blind their ballots. The signature of the blinded ballots (***Ra<bbi>***) would be present in the admin's list. The inverse blinding function ***b' (Ra<bbi>, ri)*** is applied on this signature to obtain the signature of the valid ballots. Only these ballots are considered for counting. *In Sensus system, since the counter checks that every ballot is unique, the first scenario would be detected and rejected at voting phase itself. Also, duplicates could be identified and discarded from the list of published ballots after the collection phase.*
  - *Admin does foul play, creates a dummy ballot and signature (bi', Ra<bi'>) and sends to Counter.*
    This can be caught using the same procedure as mentioned above if no voter abstains from voting. The admin's list wouldn't have a signature for the blinded version of the dummy vote (Ra<bbi'>). But if some registered voters don't participate, the admin can misuse the ID and place a blinded version of the dummy vote and its signature on the list published after administration phase.
  - *A dishonest voter creates a signature of his own (Ra'<bi>) and sends (bi , Ra'<bi>) to Counter.*

- *Voter tries to change his vote during opening phase.*
  The first case can be caught since counter verifies that the signature on the ballot is that chosen by the authority for this election. In this case, Ba Ra'<bi> wouldn't be equal to bi and hence this ballot would be rejected.
  The second case isn't possible due to the Binding property of bit-commitment being used. *In Sensus system, to prevent voter from falsely accusing counter of foul play, he is asked to distribute his secret key pair to multiple trusted parties.*
  This also satisfies the property of **SOUNDNESS** mentioned in the main paper [1]. No dishonest voter can disrupt the voting.
- *Admin does foul play and doesn't supply correct signature on the blinded ballot (in Administration Phase) <or> the signature is tampered by an adversary on its way back to the voter. In these two cases, a valid ballot would get rejected by Counter due to invalid/corrupted certificate.*
  This can be caught by voter. When he gets back the admin's certificate he checks if the signature provided on the ballot is correct. If not, he announces on bulletin board the un-blinded ballot and signature pair (bi, Ra<bi>) to
  prove that admin has supplied invalid signature. Since at this point, it can't be determined if the voter is lying or not, no action is taken after administration phase and the other phases are completed omitting this disputed vote.
  In the end, if the disputed votes don't make a difference to the final result, they are ignored. If not, the election is restarted.

- **INVULNERABILITY (encompasses Eligibility and Un-reusability):** Only votes of eligible and registered candidates should be considered. Every voter should be allowed to vote only once.
  Admin checks IDi with the list of registered voters for eligibility before providing certificate. It also authenticates the signature of the voter to ensure that no impersonation or tampering took place en-route. Counter verifies that the ballot received has the authority's signature for this election. A voter can't forge admin's signature. *In Sensus system, since it is checked that the ballot received is unique, a voter can't keep replaying his ballots.*
- **PRIVACY:** All votes must be a secret. Nobody can link votes to voters.
  The channel between voters and counter is anonymous. Authority can't link ballots (bi) to their blinded versions (bbi) hence cant associate published ballots to the voter IDs sending them. The blinded ballots arriving at admin's end and the un-blinded ballots arriving at counter's end are not in temporal order (assumption 2). Hence even if admin and counter collude privacy of voters can't be compromised. When voters claim fraud by admin or counter, they only release the ballots (encrypted versions of the votes) and the not the votes directly.
- **FAIRNESS:** Nobody knows the votes, trend of voting or intermediate results before final counting.
  Only the voter has the secret decryption key to reveal the ballot. Hence all validations done on the published list of ballots gives no idea about the votes until final opening and counting stage.
- In the sensus implementation, no adversary can check by mere eavesdropping if and when a particular voter obtained signature from Admin, since the tuple (IDi, blinded ballot , signature of voter) sent from voter to admin is encrypted using the public key of the admin. Similarly the certificate sent from the admin to voter is also encrypted with public key of the voter.

2.6 Strengths and Drawbacks of the scheme

- **Strengths:**
  - The scheme is scalable and efficient since computation and communication overhead is minimal.
  - Universal Acceptability of fair voting is possible. The voting process can be accepted to be fair since every property is verifiable.
  - The scheme offers flexibility of multiple-value voting (not restricted to yes/no).

- **Drawbacks:**
  - **UNIVERSAL VERIFIABILITY** not possible: As per the main paper [1] verifiability is defined as "No one can falsify the result of voting". But subsequent literature on e-voting including the paper which implements the Sensus system [2] refined this definition. A system is said to be Universally Verifiable if anyone can independently verify that all valid votes have been counted

correctly. The proposed scheme can be verified and validated only by the registered voters who participated in the election and not by anybody else.

- **RECIEPT-FREENESS** not enabled**:** The voter has information such as the random number used to blind ballots, the admin's certificate (in Sensus system he has the receipt) which he can use to prove to others that he voted in a particular way. Hence the process is not Receipt-Free and is vulnerable to vote buying and selling.
- Too much of responsibility and work assigned to voters. Voters need to be present in all phases of voting to validate and verify the correctness of the process.
- If some voters abstain from participating in the election, fraud by Administrator in casting dummy votes can't be detected and corrected.
- The scheme depends on the presence of an anonymous channel between the voter and counter to ensure privacy of voters.

3. Efficient Receipt free voting based on Homomorphic Encryption (*Report by Nupur Kulkarni*)

3.1 Introduction

There are many desirable properties of E-voting protocols out of which receipt freeness is an important one. To keep the election free of vote buying or coercion, we need to ensure that the voter is not able to prove to a third party the content of his vote. This proof for the content of vote is referred to as generating a receipt for the vote casted. If the voter is not able to generate a receipt then it means the protocol is receipt free. Benaloh and Tuinstra initiated the work in receipt free protocols for e-voting. However the protocol which they presented was not receipt free and it was possible to generate a receipt. We will now look at the first practicable receipt free voting protocol.

In most of the voting schemes, voter is the one who chooses the vote he wants to cast. He then encrypts that vote and the protocol proceeds. In such a scenario, voter can disclose the randomness he used in the encryption scheme and a receipt could be constructed for the casted vote. To avoid this situation, in this protocol, a voter is presented with a list of encrypted votes out of which he chooses his vote.

The protocol executes in three phases:

1. Vote generation phase
2. Vote casting phase
3. Tallying phase

3.2 The Protocol:

3.2.1 Notations and definitions:

Let M denote total number of voters and N denote total number of authorities conducting the voting. It is a voting scheme with multiple values for valid votes. The total number of valid votes is L. A probabilistic encryption function E is used for encrypting the vote v such that E(v) denotes the set of encryptions for v. An encryption e for vote v is one of the values in E(v).

There are several assumptions made for the correct execution of the protocol. There is a threshold t which is minimum number of authorities guaranteed to remain honest. For communication, a bulletin board is used by the entities in the system. Everyone can read the contents posted on the bulletin board. No one can delete the contents of bulletin board and only the participants can write to it in their own sections. There exists a one way untappable channel from authorities to voters. Voters can only receive data from it however cannot prove to the third party about particular content that they received over it. No entity can listen to the communication over untappable

channel. A public list $e_1(0)\ldots e_L(0)$ of encrypted values of all valid votes is available which acts as the input for authority 1.

3.2.2 Vote generation phase:

Each authority A(i) takes as input a previous list $e_1(i-1)\ldots e_L(i-1)$. It performs re-encryption on every encrypted vote in the list. It permutes the order of votes in the list and outputs the new list constructed. This list is presented to the voter over the untappable channel. This list is accompanied by a proof of correctness for the voter to verify. Each authority generates a proof that they have performed a re-encryption correctly and for every element in the previous list, there exists a re-encryption in the new list that they have generated. The proof is only verifiable by the voter since he is the designated verifier for the particular proof. Refer to appendix for the details about the proof.

The voter can accept or decline the proof depending on its correctness. For maintaining the correctness of the protocol, a voter can decline at most N-t such proofs since minimum t authorities are guaranteed to remain honest. If a voter declines a proof, he publicly announces it and due to this, the particular re-encryption and permutation done by that authority is ignored. Therefore, the next authority in the chain takes as input the list taken as input by "this" authority ignoring this authority's computation.

ElGamal encryption scheme is used for re-encrypting values. Refer to section 2.6 for details.

The vote generation phase is shown in the Figure 4.

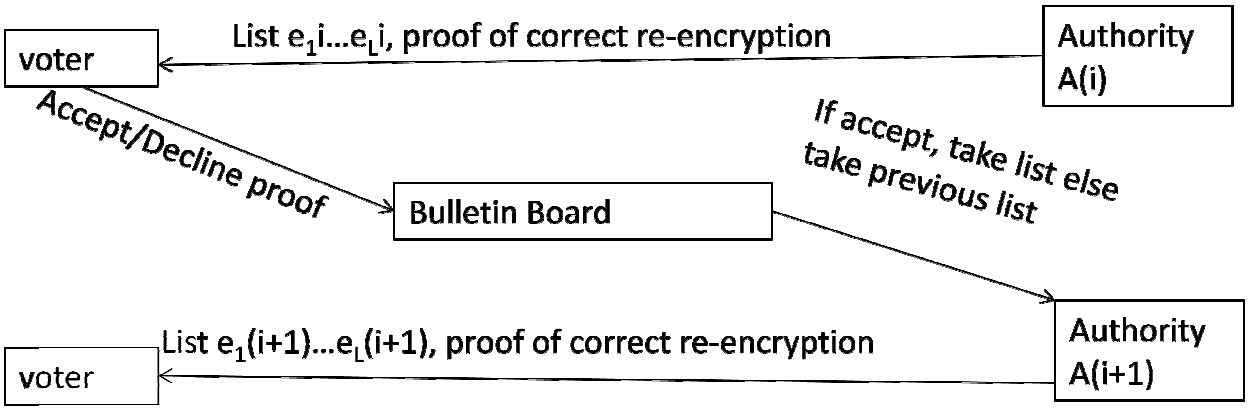

Figure 4

3.2.3 Vote Casting phase:

The voter has already accepted the proofs from the authorities for the correct re-encryption in the generation phase. Hence for casting his vote, the voter is required to simply announce the position of the encrypted vote from the list. Figure 5 shows the vote casting phase.

Figure 5

3.2.4 Tallying phase:

In the tallying phase, the encrypted values for casted votes are available on the bulletin board. These values are summed up using the homomorphic property of the ElGamal scheme. (Refer to section 2.6 for details). This addition gives the encrypted value for the sum T of votes. The decryption is performed on this encrypted sum to obtain the sum value T. The authorities come together to decrypt this value and present the proof of correctness for the decryption on the bulletin board. This proof is verifiable by everyone.

3.2.5 Insights on the protocol

The verifiability of the proof for correct decryption makes the protocol universally verifiable.

Most of the other schemes need a decryption on every single vote casted. This protocol only requires a decryption on the sum value which avoids the cost incurred in the other decryptions. Avoiding this significant overhead makes this scheme efficient.

The correctness of the protocol cannot be guaranteed with single authority in the model. If there is only one authority, the voter cannot be sure of the encrypted values of votes presented to him. A malicious authority can present any garbage to the voter and make him believe that the list is correct. The correctness is based on the fact that multiple authorities present different lists to the voter along with the proofs of correctness. Hence the voter is guaranteed about the encrypted values in the lists. The model has multiple authorities to help avoid collusion given the threshold t assumption.

3.2.6 ElGamal encryption scheme

ElGamal encryption uses a cyclic group **Zq** which is of order of a large prime number **q**. Let $\mathbf{g_1}$ and $\mathbf{g_2}$ be two generators for the group. The key pair of public key **B** and secret key **R** is constructed in such a way that every authority has their share of $\mathbf{R_i}$ in a (t,N) threshold secret sharing scheme. Every authority is publicly committed to their share of public key computed by $\mathbf{g_1}^{\mathbf{Ri}}$ .Refer to [12, CGS91] for more details. The public key is computed by $B = g_1^R$ .A set of encrypted values for vote v is given by

$$E(v) = (g_1^{\alpha}, g_2^{v} B^{\alpha})$$

Where α is random number from the group.

ElGamal encryption is used in this protocol to ensure receipt freeness. It satisfies the homomorphic property which means addition of two encryptions yield a new encryption. Given two encryptions e1=(c1,c2) and e2=(c3,c4) where c1,c2,c3,c4 are cipher blocks of ElGamal, their homomorphic addition is defined as

$$e1 \text{ XOR } e2 = (c1c3, c2c4)$$

If e1 is one of the encryptions of vote v1 and e2 is one of the encryptions for vote v2 then, (e1 XOR e2) belongs to the set of one of the encryptions of (v1+v2). This property allows the scheme to only decrypt the sum vote T. There remains no need to decrypt every single vote casted.

The decryption of the sum vote cannot be performed by a single authority. For decrypting the sum vote T from the given encryption e=(c1,c2), each authority computes the value $c1' = c1^{Ri}$ where R_i is their share of secret key. The value $c1'=c1^R$ is computed from all these values. This value can only be computed if at least t authorities participate in computing it. Refer [12,13] for more details. From the value c1', following can be computed c2/c1'. This computation can be expanded as

$$c2/c1' = g_2^T . B^{\alpha} / g_1^{\alpha R}$$

Here the terms get cancelled and the decrypted message g_2^T is obtained. Since c1' cannot be calculated with t-1 authorities, the protocol works correctly only if t authorities remain honest throughout the protocol execution.

For the homomorphic property of ElGamal encryption to be useful, we should be able to find the number of votes casted for every valid value from the decrypted sum value T. Encoding of votes is done to achieve this. Votes are encoded in such a way that given the sum value and total number of votes casted, it is possible to find out number of votes casted for each value. One way to encode the votes is to have set of valid votes as $\{1,M,M^2,\ldots,M^{L-1}\}$ where M is number of voters and L is total number of valid votes.

3.2.7 Properties of the protocol

This protocol satisfies following properties:

- Receipt-freeness

Voter is not able to generate receipt of any form proving the content of the vote he casted. Thus this could be used to prevent vote buying during voting.

- Privacy

No entity can see the values of the votes since they are encrypted. The mapping between votes and voters is not exposed either and hence privacy is ensured for every voter.

- Vote and go

Voters are not required in any other phase than voting and hence it is a vote and go type of voting scheme. It is the ideal scenario from the practicality perspective.

- Fairness

No one can predict and view the votes while voting is still in progress. Hence even if voting process is discontinued before completion, the trend of voting is exposed and it cannot adversely affect the election in future.

- Universal verifiability

Correctness of the sum of votes is publicly verifiable. The authorities post the proof of correctness for the decrypted value on the bulletin board which any one can verify.

4. A Re-encryption Mix Network Voting Scheme *(Report by Guillermo Cabrera)*

The following paragraphs will first briefly describe the background of the voting scheme presented in this section, then; the details of the three phases involved in the voting protocol will be explained. Finally, a simple example and discussion will also be provided, in order to better understand a voting scheme using mix networks.

The "All/Nothing" election scheme proposed by Park, Itoh and Kurosawa is based on initial work by David Chaum in 1981 [14]. In his paper, he describes an anonymous channel that uses a set of intermediary computers (mixes) to partially decrypt and route messages from sender to receiver, while maintaining sender anonymity. In this anonymous channel each mix will accept a ciphertext as input and will output the decryption of it. There are other types of mix networks where instead of decryption, each mix will re-encrypt the ciphertext provided as input and later all mixes will participate in the decryption.

It is the latter type of mix net (re-encryption) that is adopted and implemented making use of El Gamal encryption. Using this probabilistic (multiple representations for a single plaintext) encryption scheme will remove the problem of computing and sending long ciphertexts to the mix network; in a regular "Decryption" mix net, the size of the initial message is proportional to the number of mixes, whereas, in a "Re-encryption" mix net, the message size is irrelevant to the number of mixes.

Before explaining the protocol, it is important to note, that it is a "Yes/No" scheme, in other words, this protocol would have to be executed for each candidate in the election, most recent publications have tried to depart from this idea, and move to a "multi-value" scheme, where a vote can be casted on several candidates using a single ballot.

4.1 Protocol Details

Let us first define the assumptions and notation that will be used henceforth:

- Notation

  Voter = V # voters = i keys: B_V & R_V
  Mixes = M # mixes = k keys: B_{MG} & R_M
  Message = vote = m
  c = a ciphertext c' = re-encryption of c

- Assumptions

  Only valid (satisfying requirements by voting laws, ex. over age 18) voters are to participate in the protocol.

- Operations

  - Encryption:

    $(c_1, c_2) := (g^r, m \times (B_{MG})^r) \bmod q$
    where g (generator) and q (prime number) are given before protocol starts, and r is any number [1, q-1]

  - Decryption:

    1. Each M computes $Z := (c_2)^{R_M}$
    2. All voters compute $c_1 / (Z_1..Z_k)$

For simplicity, both encryption and decryption operations will be denoted by: **Key< data >** (applying a key to plaintext or ciphertext), in section 4.6 a simple example making use of the expanded El Gamal notation will be given, to demonstrate how concrete operations would take place.

4.2 Protocol Setup

Prior to the first phase of the protocol, the public and private keys for the mixes need to be computed. They are computed based on information that is published by an authority which includes: a large prime number (q) and a generator (g) of the cyclic group $\mathbf{Z_q^*}$ (integers formed by multiplication and applied modulo operation with q). Given this publicly available information there are two methods in which a public and secret key can be computed:

One Mix	**Multiple Mixes**
The single mix chooses: $\mathbf{R_M}$ **:=** Any number from range {1, …, q-1} The single mix computes: $\mathbf{B_M} := g^{R_M}$ Note that the public key is B_M and not B_{MG}, meaning it is a key for a single mix and not a "group" of mixes.	Each mix chooses: $\mathbf{R_M}$:= Any number from range {1, …, q-1} Each mix computes and broadcasts: g^{R_M} Each mix computes: $\mathbf{B_{MG}} := \prod g^{R_M}$

Table 1: Choosing private and public keys for mixes

We will assume we have multiple mixes, thus, we will have a shared public key for the Mix (as a group): B_{MG}. And each mix will have its own private key R_M.

4.3 Registration phase

During this phase, voters will choose their private and public keys; send a signed copy of the public key through the mixes. The mixes will re-encrypt, shuffle and later decrypt the public keys sent by voters and publish them for all voters to see. This process is creating a list of digital pseudonyms, which will be used in subsequent phases to verify signatures made by the anonymous voters. The three main steps are:

1. Each V_i chooses B_{Vi} and R_{Vi} and sends $B_{MG}< B_{Vi} \bullet R_{Vi} < B_{Vi} >>$ to the mixes.
2. k mixes re-encrypt, shuffle and decrypt $c:= B_{MG}< B_{Vi} \bullet R_{Vi} < B_{Vi} >>$

   M_1: $B_{MG}< c >$
   M_2: $B_{MG}< B_{MG}< c > >$
   ...
   M_k: $B_{MG}< \dots B_{MG}< B_{MG} <c>> \dots >$

   After all mixes have participated in the re-encryption process, we will end with a ciphertext for each voter similar to

   $c':= B_{MG}< \dots B_{MG}< B_{MG} <c>> \dots >$

   At this point, each voter's ciphertext will be decrypted by applying each of the mixes private keys and once M_k obtains the plaintext p, it will apply the voter's public key (attached in message) to the part encrypted using the voter's private key (R_{Vi}) in order to validate the integrity of the message and make sure that it is a correct public key.

   M_1: $R_{M1}< c' >$
   M_2: $R_{M2}< R_{M1}< c' >>$
   …
   M_k: $R_{Mk}< \dots R_{M2}< R_{M1}< c' >> \dots >$
   $p:= B_{Vi} \bullet R_{Vi} < B_{Vi} >$

In the end, all public keys for all voters (B_{Vi}) will be posted in lexicographical order to a public broadcast channel. Having all keys posted in a specific order prevents any association to be formed between the output of the mixes and the initial input sent by the voters, thus, effectively maintaining anonymity.

4.4 Claim Phase

After all voters have submitted their public key, they check the list of posted public keys received by the mixes. If a voter does not see his public key, he claims and the election process stops. Otherwise, if there is a period of time when there are no claims, the election process continues with the voting phase.

What are the reasons for a public key to not appear in the final list? There could be dishonest mixes in the network; let us assume M_k (last mix) is bad, it has access to the plaintexts and could easily rewrite the message to include some other public key of an invalid voter, it could also decide not to post a specific key. This is a violation of the

election process and could later introduce irregularities, reason why protocol should be stopped and registration recreated.

4.5 Voting Phase

The vote to be casted is computed by choosing two random numbers that create a Yes or No answer by applying an XOR operation at a bit level. If the result of applying XOR to two numbers results in a tautology (all 1 values) then the voter would have casted a Yes vote, otherwise it would be considered as a No vote. As an example, a Yes vote could be constructed out of numbers 1 (01) and 2 (10), and a No vote from 2 (10) and 3 (11).

Let us know describe the steps involved in this phase:

1. Each V_i chooses two random numbers N_{i1} and N_{i2} such that $m_i := N_{i1}$ XOR N_{i2}
2. Each V_i chooses n_{i1} and n_{12} randomly, then computes and writes:

   $(a_{i1},b_{i1}) := B_{MG}<B_{Vi} \bullet R_{vi}<N_{i1} \bullet pad>, n_{i1} >$
   $(a_{i2},b_{i2}) := B_{MG}<B_{Vi} \bullet R_{Vi}<N_{i2} \bullet pad>, n_{i2} >$

   The pad serves two purposes: it sets all the plaintexts to a specific size and at the same time acts as a confounder. This prevents any attacker from obtaining a traffic pattern or even discovering the plaintext.
3. At this point the list in the public broadcast channel contains two ciphertexts for each voters, each one of them being the encryption of a number chosen by the voter that XORed with the other number would create the vote:

   $((a_{i1},b_{i1}) , (a_{i1},b_{i1})), ((a_{i1},b_{i1}) , (a_{i1},b_{i1})), \ldots$

   Each mix will now process each ciphertext, re-encrypt it, shuffle it and then pass it to the next mix.

   For i=1, … , k
   M_i computes:
   $(a'_{j1},b'_{j1}) := B_{MG}< (a_{j1},b_{j1}), e_{j1} >$
   $(a'_{j2},b'_{j2}) := B_{MG}< (a_{j2},b_{j2}), e_{j2} >$

   for each j, where e_{j1} and e_{j1} are random numbers, M_i writes the following in lexicographical order:
   $\{((a'_{j1},b'_{j1}) , (a'_{j2},b'_{j2}))\}$

4. Once all ciphertexts have been processed, a separate mix M_0 will have the sole task of choosing a random bit d_i for each i (voter). If 0 is selected, then all mixes (M_1, …, M_k) will participate in decrypting the first ciphertext of the voter, on the other hand, if 1 is selected, then the mixes will decrypt the second ciphertext. Let us remember that each ciphertext contains a random number, and only if both are XORed will we get the vote value.
5. At this point, one of the two ciphertexts for each voter has been decrypted, let us assume M_k has the plaintext $B_{Vi} \bullet R_{vi}<N_{i1} \bullet pad>$ and publishes each plaintext's pad. Everyone checks the form of the pad of all voters, if the form is not consistent among all messages then a message could have been tampered with, meaning there was a disruption. If such a disruption occurs the protocol stops, at this point there has been no disclosure of the vote values. However, if no disruption occurred for the first batch of decrypted messages, then the remaining pieces are decrypted and the form of pad is checked again, if disruption occurs at this second round, then the protocol stops.
6. For each message i such that no disruption occurred, "m" (the vote value) is formed by applying XOR to both numbers in plaintexts:

$m_i := N_{i1}$ XOR N_{i2}

4.6 Sample Scenario

This example will show the details of the re-encryption and decryption process using the El Gamal encryption scheme in the voting phase. For simplicity we will assume there is only one voter, two mixes and we will only be working with one of the two ciphertexts containing one of the random numbers that would make up the vote value. Let us also assume that M_1 has selected its R_M to be 2 and M_2 has selected its R_M to be 3. Therefore, according to Table 1, the public key (B_{MG}) would be g^2g^3.

1. We have assumed both numbers (N_{11} and N_{12}) have already been chosen, and **m** := B_{Vi} • R_{vi}<N_{11} • pad>
2. V_1 chooses n
   (g^n, **m** x $(g^2g^3)^n$) mod q

3. M_1 chooses e1 and computes
   (g^ng^{e1}, **m** x $(g^2g^3)^n$ x $(g^2g^3)^{e1}$) mod q

   M_2 chooses e2 and computes
   ($g^ng^{e1}g^{e2}$, **m** x $(g^2g^3)^n$ x $(g^2g^3)^{e1}$ x $(g^2g^3)^{e1}$) mod q

4. This step is not relevant to this example as we only have one ciphertext.
5. M_1 computes $Z_1 := (g^ng^{e1}g^{e2})^2 = g^{2n}g^{2e1}g^{2e2}$
   M_2 computes $Z_2 := (g^ng^{e1}g^{e2})^3 = g^{3n}g^{3e1}g^{3e2}$

   The voter computes (**m** x $(g^2g^3)^n$ x $(g^2g^3)^{e1}$ x $(g^2g^3)^{e1}$) /($g^{2n}g^{2e1}g^{2e2}$ $g^{3n}g^{3e1}g^{3e2}$) which yields **m** after cancelling out factors.

4.7 Concluding Remarks

The proposed protocol enhances some aspects of the original mix network proposed in [14]. Since the goal of electronic voting is managing voting through the internet, most devices would rely on regular network communications, whereas, mixes would have a dedicated network where bandwidth might not be a concern. Therefore, having the size of initial ciphertexts not rely on the number of mixes is a great improvement. The protocol satisfies the following properties:

- Accuracy: Votes may be modified by any mix which is dishonest, however, the final count of votes will not take into account the tampered vote, thus, effectively counting the final set of votes.
- Fairness: As the name of the protocol suggests, either all votes are processed or none at all, thus, if some disruption occurs, no votes will be disclosed, so that if a re-election were to organized, voters would not be biased by looking at results of previous elections.
- Robustness: The system is able to perform in the event mixes fail or are dishonest, only a single honest mix is necessary for anonymity and privacy to be preserved.
- Receipt-free: No type of receipt is given to voter for him to share his value while voting. Recall that voters just check the form of pads to verify their vote being counted, but the system does not provide him with the copy of the numbers the voter initially chose to make up his vote.
- Privacy: Achieved via the mix network and re-encryption.
- Verifiable: Since each voter is able to verify that his vote was counted, yet, it is not universally verifiable, since none voters cannot take part in the verification process.
- Invulnerability: only eligible voters are permitted, these are screened by an authority previous to the registration phase in this protocol.

5. Conclusion

In previous sections we have seen the three main schemes that are used in electronic voting, from table 2 we can see how each scheme satisfies a specific set of properties. These properties have been defined based on the requirements that a regular election process would demand. As mentioned earlier in the introduction section, these electronic protocols add several levels of complexity to the election process: To begin with, it is possible to manipulate more votes electronically than it is if we had punch cards or bubble in ballots, also, computer systems and networks are subject to a plethora of attacks that might disrupt individual votes or the entire election process.

The work presented in this report constitutes initial work performed in the field of electronic voting; focus has been given to early protocols that have addressed specific problems (ex. universal verifiability, robustness). Most of the later work in voting schemes is built upon the idea of blinding signatures, homorphic encryption or mix networks; many authors have tried to develop a protocol capable of satisfying all the requirements listed in table 2, yet, most attempts end in tradeoffs involving some of these properties.

Given the complexity in achieving these properties recent work in [16] has proposed a model to quantify these properties and define how close a system is to being perfect with respect to each of the requirements. Furthermore, given the wide array of voting system implementations and the various requirements by different governments, work is also being done in terms of standardizing basic requirements for electronic voting, the idea behind this work would define a basic set of required security requirements based on a couple of assumptions, which needs to be extended depending on the particular election the system would be used for.

Finally, since the main goal for electronic voting is concentrated on internet voting instead of kiosk voting, it is imperative that the security of voter's computing devices be guaranteed. This poses a difficult task given the different number of threats on these systems; moreover, having the ability to cast a vote remotely adds the possibility of coercion, in which a voter is forced to vote for a specific candidate. Nonetheless, this gives us a scope of the type of work necessary in this field.

Electronic voting has a great potential, and is an inevitable step in future years to come, comments of early adopters of these systems has proved valuable in driving the direction in which these systems should be headed at; problems during election processes by these early adopters have opened new lines of research within the field. It is a matter of time before these systems are completely adopted, however, educating citizens on the security and use of these systems is critical, in order to reach a point where we completely rely on them.

	Accuracy	Fairness	Privacy	Invulnerability	Verifiability	Receipt-Freeness	Convenience	Flexibility	Robustness
Blind Signatures	X	X	X	X				X	
Homorphic Encryption	X	X	X	X	X	X	X	X	
Mix Networks	X	X	X	X	X	X			X

Table 2: Summary of e-voting protocols and properties satisfied

6. References

[1] A. Fujioka, T. Okamoto, and K. Ohta. A practical secret voting scheme for large scale election. In *AUSCRYPT 92*, pages 244–251. Springer-Verlag, 1992. LNCS 718.

[2] Lorrie Faith Cranor and Ron K. Cytron, Sensus: A Security-Conscious Electronic Polling System for the Internet. Proceedings of the Hawai`i International Conference on System Sciences , January 7-10, 1997, Wailea, Hawaii, USA.

[3] Chaum, D. Blind signatures for untraceable payments. In Proceedings of Crypto 82, Plenum Press, New York. 1983, pp. 199-203.

[4] Rivest, R.; A. Shamir; L. Adleman (1978). "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems". Communications of the ACM 21 (2): pp.120–126. doi:10.1145/359340.359342

[5] D. Chaum, "Security without Identification: Transaction Systems to Make Big Brother Obsolete", Communications of the ACM, Vol.28, No.10, pp.1030-1044 (Oct., 1985).

[6] O. Goldreich, S. Micali, and A. Wigderson, "How to Play Any Mental Game or a Completeness Theorem for Protocols with Honest Majority", Proceedings of the 19th Annual ACM Symposium on Theory of Computing, pp.218-229 (May, 1987).

[7] M. Naor, "Bit Commitment Using Pseudo-Randomness', in Advances in Cryptology -- CRYPTO '89, Lecture Notes in Computer Science 435,Springer-Verlag, Berlin, pp.128-136 (1990).

[8] K. Ohta, "An Electrical Voting Scheme using a Single Administrator" (in Japanese), 1988 Spring National Convention Record, IEICE.

[9] Nurmi, H., Salomaa, A., and Santean, L. Secret ballot elections in computer networks. Computers and Security, 36, 10 (1991), pp. 553-560.

[10] Electronic voting: computerized polls may save money, protect privacy, LF Cranor - Crossroads, 1996 - portal.acm.org.

[11] Martin Hirt and Kazue Sako. “Efficient Receipt-Free Voting Based on Homomorphic Encryption”

[12] Torben P. Pedersen. A threshold cryptosystem without a trusted party (extended abstract).

[13] Ronald Cramer, Rosario Gennaro, and Berry Schoenmakers. A secure and optimally efficient multi-authority election scheme.

[14] Chaum, D.L.: Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. Communications of the ACM, Vol. 24, No.2, (1981), 84-88

[15] C. Park, K. Itoh and K. Kurosawa: Efficient Anonymous Channel and All/Nothing Election Scheme. Advances in Cryptology - EUROCRYPT ’93, LNCS 765, pp. 248-259, 1994.

[16] D. Chaum, M. Kutylowski, R. L. Rivest, P. Y. A. Ryan: 07311 Abstracts Collection -- Frontiers of Electronic Voting. 2008

7. Appendix

7.1 Blind RSA signatures:

To encrypt a message m using the RSA encryption scheme, it is raised to the power of a public exponent e, and then the remainder when the result is divided by a publicly specified value N (N is the product of two large prime numbers p and q) is taken [4]. *This would be the equivalent operation performed when a message is to be encrypted with the public key Ba of an administrator using the RSA signing scheme.*

$$m' = Ba\langle m\rangle = m^e \bmod N \qquad (1)$$

To decrypt m’, the same procedure is adopted but m’ is raised to the power of a private/secret exponent d. The private exponent d and the public exponent e are RSA keys satisfying the property $e \times d = 1 \pmod{(p-1)(q-1)}$ (x is multiplication operation) [4]. *This would be the equivalent operation performed when a message encrypted with the public key Ba using the RSA encryption scheme, needs to be decrypted by applying the private key Ra of the administrator.*

$$Ra\langle m'\rangle = m'^d \bmod N \qquad (2)$$
$$= (m^{ed} \bmod N) = m \bmod N = m \text{ (m is a value between 0 and N -1)}$$

To blind a message m using the RSA signing scheme, a random number r is chosen and encrypted using the RSA encryption method described above in (1). The result is then multiplied with the message m.

$$b = m \times r^e \bmod N = m * Ba\langle r\rangle \qquad (3)$$

The blinded message b is then sent to the administrator or authority for signature. The authority supplies the signature by applying its private key Ra on the received message b using the RSA decryption method described above in (2).

$$Ra\langle b\rangle = b^d \bmod N = (m \times r^e \bmod N)^d \bmod N = (m'^d \times r^{ed} \bmod N) \bmod N$$
$$= m'^d \bmod N \times r \bmod N = Ra\langle m\rangle \times r \qquad (4)$$

This signature is then sent back to the message author, who divides the signature by r mod N to retrieve the original message m with admin’s signature on it.

$$Ra\langle b\rangle / r = (m'^d \bmod N \times r \bmod N) / r \bmod N = m'^d \bmod N = Ra\langle m\rangle$$