available at www.sciencedirect.com

SCIENCE DIRECT®

journal homepage: www.elsevier.com/locate/cose

Computers & Security

A framework and taxonomy for comparison of electronic voting schemes

*Krishna Sampigethaya, Radha Poovendran**

Department of Electrical Engineering, University of Washington, Campus Box 352500, Paul Allen Center, AE 100R, Seattle, WA 98195-2500, USA

ARTICLE INFO

Article history:
Received 29 September 2004
Revised 8 November 2005
Accepted 8 November 2005

Keywords:
Electronic voting
Cryptography
Protocols
Number theory
Elections
Secret ballot
Mixnets

ABSTRACT

Electronic voting is an emerging social application of cryptographic protocols. A vast amount of literature on electronic voting has been developed over the last two decades. In this paper, we provide a framework that classifies these approaches and defines a set of metrics under which their properties can be compared. Such a methodology reveals important differences in security properties between the classes and allows for selection and future design of voting schemes, based on application requirements. We illustrate the use of our framework by analyzing some of the existing electronic voting schemes.

1. Introduction

Voting is a process at the heart of a democratic society. Voting schemes have evolved from counting hands in early days, to systems that include paper, punch card, mechanical lever, and optical-scan machines. Recent democratic elections using voting machines have shown that the winning margins could be less than the error margins of the voting systems themselves, making election an error prone task. Use of electronic voting has the potential to reduce or remove unwanted human errors. In addition to its reliability, e-voting can handle multiple modalities (such as voice assistance for handicap), and provide better scalability for large elections. E-Voting is also an excellent mechanism that doesn't require geographical proximity of the voters. For example, soldiers abroad can participate in elections by voting online.

While e-voting has been an active area of research for the past two decades, efforts to develop real-world solutions have just begun (Caltech-MIT, 2001; Gritzalis, 2002a), posing several new challenges. The use of insecure Internet, well documented cases of incorrect implementations, and the resulting security breaches have been reported recently (Jefferson et al., 2004; Kohno et al., 2004). These challenges and concerns have to be resolved in order to create public trust in e-voting.

An important step towards streamlining this effort is to develop a framework and identify necessary properties that a secure and trusted e-voting system must satisfy to reduce discovery redundancy. Such a framework will allow us to evaluate as well as compare the merits of existing and future candidate e-voting schemes. In this paper we provide such a framework. To maintain clarity, we minimize the use of formulae and emphasize on main concepts.

* *Corresponding author*. Tel.: +1 206 221 6512; fax: +1 206 543 3842.
E-mail addresses: rkrishna@ee.washington.edu (K. Sampigethaya), radha@ee.washington.edu (R. Poovendran).
0167-4048/$ – see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cose.2005.11.003

This paper is organized as follows. The next section describes the e-voting model and the properties that have to be satisfied by voting schemes, and the section following the next introduces necessary cryptographic building blocks used by voting schemes. In Section 1 framework for classification of voting schemes, our unified framework for classification of e-voting schemes is described. Section 5 covers hidden voter class, and the subsequent two sections cover hidden vote class and the hidden voter with hidden vote class, respectively. Section 8 compares the approaches described in Sections 5 to 7. The last section presents our conclusions and suggestions.

2. Description of voting model

A generic voting model consists of the following entities: Voter, Authority, Candidate, and Adversary. *Voters* who are eligible, will vote to choose among the contesting *candidates*. These candidates may be pre-specified, or write-in choices of voters. Often, how someone votes has to be kept private and the final count has to be reliable and verifiable. Equally important is the ability to prevent inference of the partial tally when the voting is in progress. An *authority* is an entity, responsible for conducting the election. An *adversary* is a malicious entity in the voting model, which attempts to manipulate the voting and/or tallying. An *external adversary* may actively try to coerce a voter or buy a voter, and may passively try to breach the privacy of voters. An *internal adversary*, apart from breaching privacy, may try to modify or reveal the partial tally as well as corrupt the authority.

2.1. Voting scheme stages

A voting scheme generally consists of *five* stages as shown in Fig. 1. Each stage will have the entities in the voting model, participating in a cryptographic protocol to achieve specific requirements. All the stages follow in order, except for the verification stage which can be used multiple times to enforce protocols. We now present a list of requirements that have to be addressed by a voting scheme and protocols that implement it.

2.2. Requirements of voting schemes

In order to be deployed widely, a voting scheme is expected to satisfy certain *general security* requirements determined by the application (Gritzalis, 2002b), and also some *system implementation* specific requirements. To be considered secure against adversarial attacks a voting scheme must satisfy additional security requirements. However, as shown in Fig. 2, some of the requirements turn out to be conflicting, and tradeoffs often arise in system design. We now present these requirements by categorizing them as general security, adversary counter-attack, and system implementation requirements.

(1) General Security Requirements:
   (a) *Eligibility*: In any voting scheme, only valid voters who meet certain pre-determined criterion are eligible to vote. Ability to verify voter's validity and a mechanism to ensure that each entity can cast permitted number of votes, is a must for a voting scheme.
   (b) *Privacy*: In a secret ballot, a vote must not identify a voter and any traceability between the voter and its vote must be removed. *Maximal privacy* is achieved by a voting scheme, if the privacy of a voter is breached only with a collusion of all remaining entities (voters and authorities).
   (c) *Verifiability*: A voter should be able to verify if its vote was correctly recorded and accounted for in the final vote tally. There are two flavors of this requirement. One is the *individual verifiability* (Sako and Killian, 1995) where only the voter can verify its vote in the tally. The second is *universal verifiability* (Sako and Killian, 1995) where after the tally is published, anyone can verify that all valid votes were included, and the tally process was accurate. Universal verifiability is more practical since assuming voters to verify their votes individually is not realistic. Verifiability requirement needs voter to be linked to vote, and hence is in contradiction to privacy. However, this requirement is crucial in gaining trust of the voter in the voting system.
   (d) *Dispute-freeness*: Any voting scheme must provide a mechanism to resolve all disputes in any stage. The notion of universal verifiability is similar but limited to the voting and tallying stages. As will be shown later, satisfying dispute-freeness can make design of schemes complicated.
   (e) *Accuracy*: Voting schemes must be error-free. The votes must be correctly recorded and tallied. Votes of invalid voters should not be counted in the tally. Universal verifiability property is directly related to accuracy.

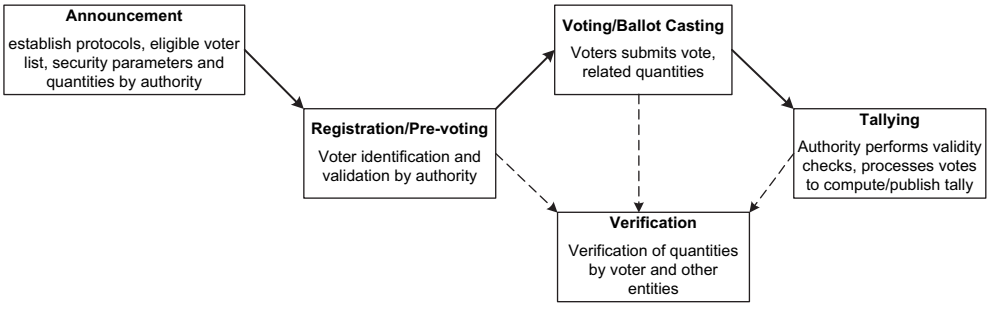

Fig. 1 – A schematic diagram indicating various stages of a voting scheme. Note that more than one stage may require different forms of verification.

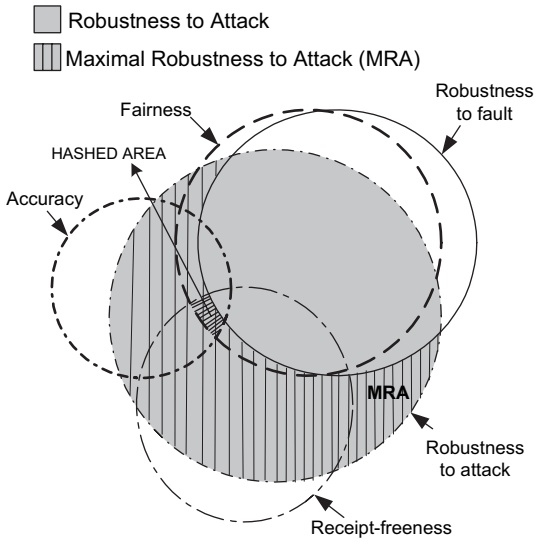

Fig. 2 – A Venn diagram illustrating the relationships between some security properties of voting schemes. The intersecting regions determine the properties satisfied and are the design spaces for voting schemes. As an example, the indicated hashed area is the design space for a scheme that *can satisfy* accuracy, fairness, receipt-freeness and maximum robustness to attack, but *cannot satisfy* robustness to fault property.

(f) *Long-term privacy*: In periodic elections, long-term untraceability or privacy may have to be provided to the voter. Information-theoretically secure cryptographic schemes (Stinson, 2002) are used to satisfy this property.

(g) *Fairness*: In order to conduct an impartial election, no one should be able to compute the partial tally as the election progresses.

(2) Adversary Counter-attack Requirements:Apart from general security features, a voting scheme must be resilient to attacks by an adversary described in the voting model. To ensure resilience, the following requirements have to be met.

(a) *Robustness*: A scheme has to be robust against active or passive attacks (corrupt authorities/voters) as well as faults (non-participating authority/voters). A voting scheme achieving maximum robustness in the presence of corrupt authorities, requires a collusion of all authorities to disrupt the election. But this also necessitates *all* the authorities to participate in conducting the election. Any non-participating authority can also disrupt the election, leading to zero robustness to faults. Hence, as shown in Fig. 2, an inherent conflict between robustness to adversary and robustness to fault exists in a voting scheme.

(b) *Receipt-freeness*: A voter should not be provided with a receipt with which it may be able to prove vote to any other entity. Receipt-freeness (Benaloh and Tuinstra, 1994) has the same notion of untraceability or privacy.

(c) *Incoercibility*: An adversary may attempt to coerce a voter and manipulate the manner in which a vote is cast. An adversary may also force a voter to abstain from casting a vote, or may even represent a valid voter at *any* stage of the voting scheme, by obtaining the voter's private key. An incoercible voting scheme, will not allow such an adversary to coerce voters.

(3) System Implementation Requirements:A voting scheme satisfying the above requirements must also be implementable. In particular, the scheme must satisfy at least the following requirements:

(a) *Scalability*: The complexity of the protocols used in a voting scheme, is a major factor in its practical implementation. An efficient voting scheme has to be scalable with respect to storage, computation, and communication needs as a fraction of the number of voters.

(b) *Practicality*: A practical voting scheme should not have assumptions and requirements that may be difficult to implement on a large scale.

3. Cryptographic building blocks of e-voting schemes

Before describing our framework, we provide a brief description of the cryptographic primitives and modules that constitute the protocols of a voting scheme. Notation used in this paper is presented in Table 1. Modular arithmetic is assumed for all cryptographic operations presented in our exposition (Stinson, 2002).

Table 1 – Standard notation in our presentation

Notation	Description
$\{V_1, V_2, \ldots, V_n\}$	Set of n voters
$\{v_1, v_2, \ldots, v_m\}$	Set of votes cast by voters
$\{A_1, A_2, \ldots, A_k\}$	Set of k election authorities
$E_K(m)$	Encryption of a message m with a key K
$D_K(c)$	Decryption of a ciphertext c with key K^{-1}
$K_{x_i}, K_{x_i}^{-1}$	Public key, private key of an entity x_i. Note that $D_{K_i}(E_{K_i}(m)) = m$
$\mathcal{H}(x)$	One-way hash of x
Z_n	$\{0, 1, 2, \ldots, n-1\}$
Z_n^*	$\{1, 2, \ldots, n-1\}$ where n is a prime
$x \parallel y$	Concatenation of strings x and y
$x \in_R S$	x is a random element from the set S
$\{x_j\}$	Set of elements x_j's
$\gcd(x, y)$	Greatest common divisor of x and y
$\mathrm{comm}(m, K)$	Commitment scheme for message m using a key K
$\mathrm{blind}(m, r, K)$	Blinding technique of the message m using random r and public key K
$\mathrm{sign}_P(m, K^{-1})$	Signature scheme of a participant P with private key K^{-1}

3.1. Secure channels

For interaction between entities over an insecure medium, voting schemes use the following secure communication channels:

- *Private channel* (Cramer et al., 1996) – a secure but observable communication channel that can be implemented using private key or public key cryptosystem.
- *Untappable channel* (Sako and Killian, 1995) – a *physically* secure but unobservable communication channel.

3.2. Anonymous channels

During vote submission, voter privacy can be provided by concealing the identity of the voter. This is accomplished using an *anonymous channel* – a communication channel where the voter (sender) is anonymous to the authority (receiver) and to any observer of the communication. In Chaum (1981), a multistage system consisting of cryptography and shuffling/permutations, called *mixnet*, was proposed as an anonymous channel. We describe it below.

3.2.1. Mixnets

Fig. 3 presents a schematic diagram of a generic mixnet. At stage i(Mix_i), a batch of inputs are received and transformed using either decryption or encryption, permuted and parallely transferred to stage $i+1$. Based on the cryptographic transformation used, the mixnet is called *decryption mixnet* or *re-encryption mixnet*. We describe them below.

3.2.1.1. *Decryption mixnet*. Let K_i be the public key of the ith stage. Let the sender of the mixnet be a voter and the receiver be an authority, in a voting scheme. A sender V_j concatenates a message m_j with a random string r_j as $(m_j \parallel r_j)$, then encrypts as, $E_{K_1}(E_{K_2}(\ldots(E_{K_l}(m_j \parallel r_j))\ldots))$ and broadcasts it. Mix_i with private key K_i^{-1}, receives inputs as, $E_{K_i}(E_{K_{i+1}}(\ldots(E_{K_l}(m_j \parallel r_j))\ldots))$ from Mix_{i-1}. The mixnet algorithm can be described as follows.

Input. $E_{K_1}(E_{K_2}(\ldots(E_{K_l}(m_j \parallel r_j))\ldots))$; $j = 1, \ldots, n$.
For $i = 1, \ldots, l$.
For $j = 1, \ldots, n$.
Step 1. Decrypt as $D_{K_i}E_{K_i}(E_{K_{i+1}}(\ldots(E_{K_l}(m_j \parallel r_j))\ldots)) = E_{K_{i+1}}(\ldots(E_{K_l}(m_j \parallel r_j))\ldots)$.
Step 2. Lexicographically order all decrypted quantities obtained in Step 1.
Output. $\{m_j\}_R$, a batch of mixed messages that cannot be traced back to senders.

We note that RSA (Rivest et al., 1978) based mixnet requires the voter to perform l encryptions. An efficient decryption mixnet based on ElGamal cryptosystem (ElGamal, 1985), requiring only one encryption by voter was proposed in Park et al. (1994). A re-encryption mixnet also proposed in Park et al. (1994), is described below.

3.2.1.2. *Re-encryption mixnet*. A sender V_j uses random string r_j, encrypts message m_j as $E_K(m_j, r_j) = (g^{r_j}, K^{r_j}m_j)$ where $K = g^S$ is the public key of the receiver (authority), and broadcasts it. The mixnet algorithm is as follows:

Input. $E_K(m_j, r_j) = (g^{r_j}, K^{r_j}m_j)$; $j = 1, \ldots, n$.
For $i = 1, \ldots, l$.
For $j = 1, \ldots, n$.
Step 1. Re-encrypt with random string $r_{i,j}$ as:

$$E_K\left(m_j, r_j + \sum_{a=1}^{i-1} r_{a,j} + r_{i,j}\right) = \left(g^{r_j+\sum_{a=1}^{i} r_{a,j}}, K^{r_j+\sum_{a=1}^{i} r_{a,j}} m_j\right). \quad (1)$$

Step 2. Randomly permute all re-encrypted quantities obtained in Step 1.
Output.

$$\left\{E_K\left(m_j, r_j + \sum_{a=1}^{l} r_{a,j}\right) = \left(g^{r_j+\sum_{a=1}^{l} r_{a,j}}, K^{r_j+\sum_{a=1}^{l} r_{a,j}} m_j\right)\right\}_R,$$

a batch of randomized encryptions that cannot be traced back to senders.

In a mixnet, a disruption can occur when a *corrupt mix* may try to modify some of its inputs or a *faulty mix* simply fails to perform its operation. The above mixnets in their current form are not robust to such a corrupt or faulty mix (Pfitzmann, 1994; Ogata et al., 1997). By introducing verification of mixing, the mixnets of Park et al. (1994) were made robust to multiple

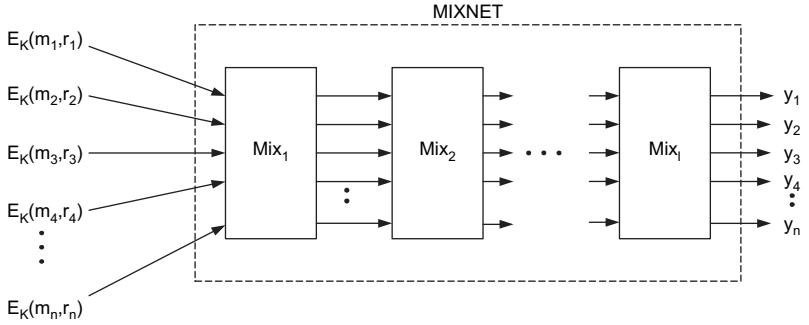

Fig. 3 – A general form of mixnet with l mixes and n inputs, $\{E_K(m_j, r_j)\}$, where $\{m_j\}$ are the messages and $\{r_j\}$ are random strings. In a *decryption mixnet*, K is public key of the mixnet, and output sequence $\{y_i\}$ contains decrypted messages $\{m_j\}_R$ in random order. In a *re-encryption mixnet*, K is the public key of the receiver, and the output sequence $\{y_i\}$ contains re-encrypted $\{E_K(m_j, r'_j)\}_R$ in random order.

corrupt mixes in Sako and Killian (1995),[1] but robustness to a faulty mix was still a problem in the modified decryption mixnet.

Approaches to improving robustness and efficiency of mixnets can be broadly classified as *optimistic* and *pessimistic*. An optimistic approach achieves the best performance when there is no disruption at any mix (Abe, 2000; Jakobsson et al., 2002; Golle et al., 2002). However, robustness is sacrificed, since a disruption requires repetition of the entire mixing, also resulting in a degradation in efficiency. On the other hand, a pessimistic approach provides verification at every stage and has no adaptive mechanism to improve efficiency when there is no disruption at any mix (Neff, 2001; Furukawa and Sako, 2001; Groth, 2003).

3.2.2. DC-net

Use of an *anonymous broadcast channel* provides maximal privacy as well as incoercibility. An anonymous broadcast channel called *DC-net* (dining cryptographers network), was proposed in Chaum (1988b). The idea is that, given n participants (voters) of the DC-net, it is impossible to trace the sender of a broadcast without a collusion of the remaining $n-1$ participants. Such a channel however, has problems with scalability and robustness, since a corrupt participant can block transmissions of honest participants without being traced and a non-participant would disrupt the channel. A recent work in Golle and Juels (2004) addresses the robustness problem of DC-nets by efficiently detecting corrupt participants.

3.3. Bulletin board

The vote and voter verifiability requirements can be achieved with a publicly accessible bulletin board. The bulletin board as defined in Cramer et al. (1997), is a public, broadcast communication channel that has memory. Any information that is broadcast will be stored in memory and readable by anyone. The bulletin board may contain designated, authenticated sections for eligible voters. An authenticated voter has write-only (append) access to its designated section. Such a bulletin board can be implemented robustly using multiple servers. In a voting scheme, each eligible voter will post a vote in its section. This allows for verification that vote is recorded correctly, and that all valid votes are included later in the tally. The authority also uses the bulletin board to post information. A special form of bulletin board will contain no designated sections, and is useful to remove link between voter and vote when achieving incoercibility property, as seen later in Section Token and homomorphic encryption based schemes.

3.4. Blind signature

Blind signature is a cryptographic protocol that can be used to anonymize the vote. As seen later in Section Token based schemes, such a protocol when combined with anonymous broadcast channel can achieve maximal privacy property. The protocol can be described as follows.

[1] We will present this approach later in Section Bulletin board based schemes.

Step 1. A voter V *blinds* its vote v using a random string r, and the public key K_A of authority A as, $\mathrm{BV} = \mathrm{blind}(v, r, K_A)$, then signs BV using its private key K_V^{-1} as, $\mathrm{sign}_V(\mathrm{BV}, K_V^{-1})$, and sends it to A.
Step 2. A verifies the validity of V (by verifying the signature with V's public key K_V), then signs BV with its private key K_A^{-1} as, $\mathrm{sign}_A(\mathrm{BV}, K_A^{-1})$, and sends it to V.
Step 3. Finally, V verifies signature of A and then unblinds (removes r) to obtain $\mathrm{sign}_A(v, K_A^{-1})$, which is the blindly signed vote v.

Such a protocol was proposed in Chaum (1984) using RSA cryptosystem, where $\mathrm{blind}(v, r, K_A) = r^{K_A} v$.

3.5. Homomorphic encryption

An encryption algorithm E_K, is said to be homomorphic, if given $E_K(m_1)$ and $E_K(m_2)$, we can obtain $E_K(m_1 \odot m_2)$ without decrypting m_1 and m_2 individually, for some operation $\odot$. The operation $\odot$ can be a modular addition ($\oplus$, additive homomorphism) or multiplication ($\otimes$, multiplicative homomorphism). RSA public key cryptosystem (Rivest et al., 1978) possesses multiplicative homomorphism, while ElGamal (1985) and Paillier (1999) cryptosystems possess additive homomorphism.

3.6. Secret sharing

A single authority trusted to conduct the election can become corrupt or faulty. Robustness can be addressed by distributing trust over multiple authorities. It then becomes necessary to also share secrets (such as a decryption key) between them. A (t, k) *threshold secret sharing scheme* (Shamir, 1979), where $t \leq k$ can be used to share a secret S between k authorities. The scheme requires a trusted party T that executes a *shared-key generation protocol*: constructs the secret key $S = K^{-1}$, publishes the public key K, and generates k shares of the secret key. Authority A_i, receives a share s_i from T over a private communication channel. To reconstruct the secret key, t or more honest authorities have to submit their shares which are then combined. The secret key is computationally protected up to a collusion of $t-1$ corrupt and $k-t$ faulty authorities. In voting schemes, T can be a third party or the voter.

In a *verifiable secret sharing scheme* (Chor et al., 1985), the trusted party T is distributedly implemented by the k authorities themselves, but requiring increased computations and communications. Only the k authorities can verify the protocol and hence any dispute may need to be resolved by a trusted third party. In a *publicly verifiable secret sharing scheme* (PVSS) (Schoenmakers, 1999), any external party can verify the correctness of the protocols at each stage of the scheme. Use of PVSS can provide dispute-freeness property as shown later in Section Voter key threshold schemes.

3.7. Interactive and non-interactive proofs

A voter may be required to prove validity of vote, and an authority may need to prove validity of a cryptographic operation, as shown in Section Hidden vote. This can be achieved

using an *interactive proof* (Goldwasser et al., 1989), a cryptographic protocol implemented by an entity P (prover) to prove knowledge of a secret to an entity V (verifier). Such protocols normally require three interactions between P and V. During first interaction P commits to a claim and sends the *commitment* to V. In the second interaction V sends a *challenge* to P. Finally in the third interaction, P computes a *response* based on the secret, the committed value, the challenge sent by V, and sends this response to V. At the end of the protocol, V either accepts or rejects the proof of P. If such a *proof* does not leak the secret then it has *zero-knowledge property* (Goldwasser et al., 1989). The protocol may be repeated to attain high probability of detecting a corrupt P. A *non-interactive proof* (Blum et al., 1991) is obtained by applying heuristics such as Fiat and Shamir (1987) to an interactive proof. The main idea is to make P first compute the commitment and also the challenge as a one-way *hash of the commitment* and send both of these quantities along with the response, to V in a single broadcast.

4. A framework for classification of voting schemes

In a secret voting scheme, voters need to privately communicate their votes towards the final tally. There will be a *tallying authority* which is responsible for receiving the votes and conducting the tallying stage. Based on how voters *submit votes* to this tallying authority, we have the following broad classification for voting schemes.

- *Hidden voter*: The voters *anonymously submit votes*.
- *Hidden vote*: The voters *openly submit encrypted votes*.
- *Hidden voter with hidden vote*: The voters *anonymously submit encrypted votes*.

We now discuss each class in detail and analyze some existing schemes under our framework.

5. Hidden voter

In this class of voting schemes, the voter remains anonymous while sending vote without encryption to the tallying authority through an *anonymous channel* such as mixnet or DC-net. To maintain accuracy of the vote, a secure communication from the voter to the *input* of the anonymous channel is required. Hence the general form of the vote from voter V_j is:

$$E_K(v_j, r_j), \tag{2}$$

where r_j is random string, K is the public key of the anonymous channel, and v_j is the vote for a pre-specified or un-specified candidate. The tallying authority and any observer, receive a set of open votes $\{v_j\}_R$ at the *output* of the anonymous channel. Hence anyone can compute the tally, $\sum_j v_j$, in a hidden voter scheme.

To ensure that the *hidden* voter is valid, there has to be some form of identification that is associated with the vote, representing a proof of the voter's validity. Based on this we can categorize hidden voter schemes as: *token based schemes*, and *bulletin board based schemes*.

5.1. Token based schemes

The identification quantity called *token*, is obtained by the voter from the authority during the registration stage of the voting scheme, implementing a *token generation protocol*. In order to ensure voter anonymity, the token has to be random and not linkable to the voter. During voting stage, voter sends token/vote over the anonymous channel to the tallying authority. A hidden voter scheme was proposed in Chaum (1981) and improved in Chaum (1988a) and Boyd (1990). We describe the voting scheme proposed in Chaum (1981).

5.1.1. Scheme in Chaum (1981)

Announcement stage: Chaum's decryption mixnet and its RSA public key cryptosystem parameters are announced, i.e. $E_K(m,r) = E_{K_1}(E_{K_2}(\ldots(E_{K_l}(m \parallel r))\ldots))$. Each voter is associated with a unique digital signature.

Registration stage:

(1) *Token generation protocol*. The eligible voter V_j generates a random K_{V_j} and $K_{V_j}^{-1}$ (RSA encryption and decryption keys) and will set, $\text{token}_j = K_{V_j}$.
(2) V_j sends token_j to Mix_1 as $E_K(\text{token}_j, r_j)$ where r_j is a random bit string, along with its digital signature on it to prove eligibility. V_j obtains a receipt of token from Mix_1.
(3) Mix_k, outputs[2] a lexicographically ordered list of voter token_j's.

Verification stage: V_j verifies token_j is received and recorded correctly.

Voting stage: V_j encrypts its vote v_j as:

$$E_K\left(\text{token}_j \parallel E_{K_{V_j}^{-1}}\left(v_j \parallel 0^l\right), r_j'\right), \tag{3}$$

and then sends a signed copy of it to Mix_1. Note that the 0^l is a large string of zero bits (of fixed length l) that ensures with high probability, detection of incorrect processing by any of the mixes. The voter will receive a receipt of vote from Mix_1. After mixing, Mix_k outputs a lexicographically ordered list of $K_{V_j} \parallel E_{K_{V_j}^{-1}}\left(v_j \parallel 0^l\right)$ on the bulletin board, from which anyone can compute tally and also detect inaccurate votes.

5.1.2. Analysis of scheme in Chaum (1981)

As seen from Tables 2 and 3, the scheme only satisfies eligibility, privacy and individual verifiability properties. Computational privacy (equivalent to breaking RSA) is provided against any external adversary. Potentially token_j of two or more voters can be the same (collision problem). A single corrupt mix can also create inaccuracy by simply modifying the quantities it gets as input. Hence the scheme is not *accurate* or *robust*. When a voter detects an inaccuracy (complaining voter), in order to protect privacy of the voter, redoing the

[2] We assume that there exists a bulletin board that contains the output of Mix_k.

Table 2 – Comparison of schemes based on general security properties

Scheme/property	Eligibility	Privacy	Verifiable	Dispute-free	Accuracy	Fair
Chaum, 1981	✓	Com	Ind	×	×	×
Chaum, 1988a	✓	Com/Max	Ind	×	×	×
Boyd, 1990	✓	Com/Max	Ind	×	×	×
Sako and Killian, 1995	✓	Com	✓	×	✓	C
Chaum, 2004	✓	Com	Ind/CU	×	C	C
Cohen and Fischer, 1985	✓	Com	✓	×	✓	×
Cohen and Yung, 1986	✓	Com	✓	×	✓	C
Benaloh, 1987	✓	Com	✓	×	✓	C
Iverson, 1992	✓	Com	Ind	×	C	C
Sako and Killian, 1994	✓	Com	✓	×	✓	C
Cramer et al., 1996	✓	Com	✓	×	✓	C
Cramer et al., 1997	✓	Com	✓	×	✓	C
Schoenmakers, 1999	✓	Com	✓	✓	✓	C
Hirt and Sako, 2000	✓	Com	✓	×	✓	C
Baudron et al., 2001	✓	Com	✓	×	✓	C
Lee and Kim, 2002	✓	Com	✓	×	✓	C
Kiayias and Yung, 2002	✓	Com/Max	✓	✓	✓	C
Damgård and Jurik, 2001	✓	Com	✓	×	✓	C
Fujioka et al., 1993	✓	Com	Ind	×	×	✓
Baraani-Dastjerdi et al., 1995	✓	Com	Ind	×	C	✓
Okamoto, 1997	✓	Com	Ind	×	×	C
Juang et al., 2002	✓	Com	Ind	×	C	C
Golle et al., 2002	✓	Com	Ind/CU	×	C	C
Lee et al., 2003	✓	Com	✓	×	✓	C
Kiayias and Yung, 2004	✓	Com	✓	×	✓	C
Juels and Jakobsson, 2002	✓	Com	Ind	×	C	C
Acquisti, 2004	✓	Com	Ind	×	C	C

✓, Satisfied; ×, not satisfied; C, conditionally satisfied; Com, computational privacy; Max, maximal privacy; Ind, individually verifiable; CU, conditionally universally verifiable.

election process is inevitable. *Fairness* property is however, lost when there is a re-election, since partial tally would have been revealed previously and may affect the decisions of voters in the re-election. Another disadvantage of Chaum (1981) is that a collusion of all the mixes would breach privacy of the voter. This weakness is addressed by the schemes in Chaum (1988a) and Boyd (1990).

Unlike Chaum (1988a), the scheme in Boyd (1990) uses computation of discrete log. But the main idea of both schemes is to employ a *blind signature* technique in the token generation protocol. As seen in Section Blind signature, at the end of the token generation protocol, V_j obtains $\text{sign}_A(v_j, K_A^{-1})$, where, K_A^{-1} is the private key of authority. V_j broadcasts this signed vote during the voting stage over the DC-net. The tally is computable by any voter that is a participant of the DC-net. The *maximal privacy* is achieved, since breaching privacy of V_j requires a collusion of all the remaining voters and the authority. However, *accuracy* and *robustness* are still problems in this approach, since the authority participating in the DC-net is able to add invalid votes for any abstainee without possible detection. As seen from Tables 2 and 3, the schemes suffer from other weaknesses (including fairness) of Chaum (1981). Also, the DC-net requires all voters to participate in the tallying stage, which makes the approach *impractical*.

We note that the token based hidden voter schemes have common weaknesses with robustness, accuracy and fairness, mainly due to the anonymous channel used. Moreover, the schemes provide voter with a receipt to prove its vote, violating receipt-freeness property. These weaknesses are addressed in the category that is presented next.

5.2. *Bulletin board based schemes*

Fig. 4 presents a bulletin board scheme. The voter uses an authenticated section on the publicly accessible bulletin board (Section Bulletin board) to post a vote, which is then mixed *verifiably* by a decryption mixnet. Tokens are not needed here since only eligible voters get access to the bulletin board. Schemes proposed in Sako and Killian (1995) and Chaum (2004) belong to this category of hidden voter schemes. We describe them below.

5.2.1. *Scheme in Sako and Killian (1995)*

Announcement stage: The public key (ElGamal cryptosystem) of the anonymous channel, $K = g^S = g^{\sum_{i=1}^{l} S_i}$ is announced with $g \in G$, where G is a unique multiplicative subgroup of Z_p^* with order q, and p, q are large primes with $q|(p-1)$, and $S \in Z_q$. Each Mix_i has public, private key as (g^{S_i}, S_i).
Registration stage: The eligible V_j registers and interacts with the l stage re-encryption mixnet. The following algorithm is executed by the mixnet.

Table 3 – Comparison of schemes based on adversarial counter-attack and system implementation properties

Scheme/property	Robust	Receipt-free	Incoercible	Scalable	Practical
Chaum, 1981	×	×	×	×	×
Chaum, 1988a	×	×	×	×	×
Boyd, 1990	×	×	×	×	×
Sako and Killian, 1995	×	✓	×	×	×
Chaum, 2004	C	✓	×	✓	C
Cohen and Fischer, 1985	×	×	×	×	×
Cohen and Yung, 1986	×	×	×	×	×
Benaloh, 1987	C	×	×	×	✓
Iverson, 1992	×	×	×	×	C
Sako and Killian, 1994	×	×	×	×	✓
Cramer et al., 1996	C	×	×	×	✓
Cramer et al., 1997	C	×	×	C	C
Schoenmakers, 1999	C	×	×	×	✓
Hirt and Sako, 2000	C	✓	×	×	C
Baudron et al., 2001	C	✓	×	C	C
Lee and Kim, 2002	C	✓	×	✓	×
Kiayias and Yung, 2002	×	×	×	×	✓
Damgård and Jurik, 2001	C	✓	×	C	C
Fujioka et al., 1993	×	×	×	✓	×
Baraani-Dastjerdi et al., 1995	C	×	×	×	×
Okamoto, 1997	C	✓	×	×	×
Juang et al., 2002	C	×	×	✓	✓
Golle et al., 2002	C	×	×	C	✓
Lee et al., 2003	C	✓	×	C	×
Kiayias and Yung, 2004	C	✓	×	C	C
Juels and Jakobsson, 2002	C	✓	C	×	×
Acquisti, 2004	C	✓	C	×	×

✓, Satisfied; ×, not satisfied; C, conditionally satisfied.

Input. $E_K(v^{(f)},0)=(g^0,K^0v^{(f)})$; $f=1,\ldots,c$, where c is the number of candidates, and $v^{(f)}$ is the vote for candidate f.

For $i=1,\ldots,l$.

For $f=1,\ldots,c$.

Step 1. Commit to V_j (using V_j's public key) a random permutation $\pi_{i,j}$.

Step 2. Re-encrypt with random string $r_{i,f}$ as:

$$E_K\left(v^{(f)},\sum_{a=1}^{i-1}r_{a,f}+r_{i,f}\right)=\left(g^{\sum_{a=1}^{i}r_{a,f}},K^{\sum_{a=1}^{i}r_{a,f}}v^{(f)}\right). \quad (4)$$

Step 3. Randomly permute using $\pi_{i,j}$, all re-encrypted quantities obtained in Step 2.

Step 4. Post non-interactive proof of correct re-encryption and permutation, $\text{proof}_{i,j}$, on the bulletin board.

Step 5. Decommit $\pi_{i,j}$ to the voter V_j over an *untappable channel*, that verifies $\pi_{i,j}$, using $\text{proof}_{i,j}$.

Output. $\left\{E_K\left(v^{(f)},\sum_{a=1}^{l}r_{a,f}\right)\right\}_R$, a batch of randomized c encryptions that can be traced back to actual votes, *only* by voter V_j.

Voting stage:

(1) *Vost casting*. V_j *chooses* one of the c ElGamal encryptions from the output of the re-encryption mixnet, as its vote $E_K(v_j,r_j)$.

(2) *Mixing*. All voters finish choosing their encrypted votes, the votes are then sent through the *decryption mixnet*. Each Mix posts non-interactive proofs of correct mixing (decryption and permutation) on the bulletin board. Mix_i will output the decrypted v_j's on the bulletin board (and also the proofs).

5.2.2. *Analysis of the scheme in Sako and Killian (1995)*

Eligibility and privacy properties are satisfied, but more importantly *fairness*, *accuracy*, and *universal verifiability* properties are also achieved. The verifiable mixnet together with the publicly accessible bulletin board, provides universal verifiability property. Hence the scheme also satisfies accuracy. Fairness and robustness are achieved up to $l-1$ corrupt mixes. *Receipt-freeness* property is satisfied assuming one-way untappable channels, since the voter cannot prove its vote to adversary. Use of the untappable channels however, creates the possibility for disputes between the voter and the authority, over a communication and also makes the scheme less *practical*. The multiple communications and computations for verification required by the voter makes the scheme *unscalable* as well as *inefficient*. However, the main disadvantage of the scheme is lack of *robustness* to a faulty mix, since a single non-participating mix in the decryption mixnet can disrupt the election. Recently, a hidden voter scheme was proposed in Chaum (2004) using a robust decryption mixnet that was developed in Jakobsson et al. (2002). Receipt-freeness is innovatively achieved in Chaum (2004) using visual cryptography. But the mixnet is only *conditionally universally verifiable* (Jakobsson et al., 2002) and can produce inaccuracies. Any inaccuracy may lead to an unfair re-election as in other hidden voter schemes.

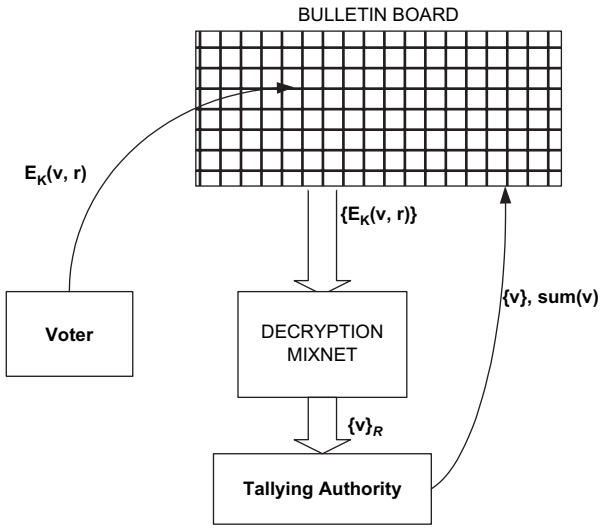

Fig. 4 – A typical bulletin board with decryption mixnet based HIDDEN VOTER scheme. Voter submits encrypted vote $E_K(v, r)$, to its authenticated section in the bulletin board. After all votes have been cast, the decryption mixnet is used to open and submit votes in random order, to tallying authority. Any observer can compute the tally $\mathrm{sum}(v)$.

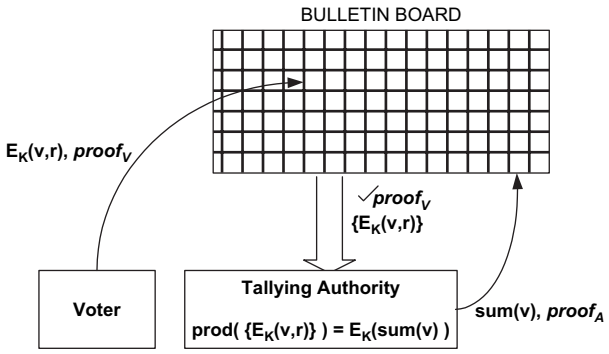

Fig. 5 – A typical HIDDEN VOTE scheme. Voter submits encrypted vote $E_K(v, r)$ and a proof_V of validity, to its authenticated section in the bulletin board. After voting stage, the tallying authority checks validity of $\{\mathrm{proof}_V\}$, collects all valid encrypted votes $\{E_K(v)\}$ and computes $E_K(\mathrm{sum}(v))$. Then tally, $\mathrm{sum}(v)$, and proof_A of correct decryption are posted on bulletin board.

6. Hidden vote

The class of hidden voter schemes could not satisfy accuracy, fairness and robustness together. These are important properties necessary for any election. Apart from unfairness, other weaknesses of the hidden voter schemes were mainly due to properties of the anonymous channels used. In order to address weaknesses of hidden voter schemes, a new genre of voting schemes called hidden vote requiring no anonymous channel, were introduced in Cohen and Fischer (1985), Cohen and Yung (1986) and Benaloh (1987).

A typical hidden vote scheme is as shown in Fig. 5. An eligible voter posts an encrypted vote in an authenticated section of a publicly accessible bulletin board. Since the voter is not anonymous, it becomes necessary for the vote to remain encrypted, to hide the link between the voter and its vote (privacy). *Homomorphic encryption* defined in Section Homomorphic encryption, provides a mechanism to directly combine the encrypted votes to get an encrypted tally. The tally is obtained by decryption. However, validity of the hidden votes has to be ensured before combining them. The voter is therefore required to provide an interactive or non-interactive proof (Section Interactive and non-interactive proofs) of validity. The general form of the vote V_j posts is:

$$\left(E_K(v_j, r_j), \mathrm{proof}_j\right), \tag{5}$$

where K is the public key of a probabilistic homomorphic encryption scheme, v_j is the vote, proof_j is a proof of validity of the vote. After verifying the proofs of voters, the tallying authority computes:

$$\prod_j E_K(v_j) = E_K\left(\sum_j v_j, \prod_j r_j\right) \text{ or } E_K\left(\sum_j v_j, \sum_j r_j\right), \tag{6}$$

due to the homomorphism of encryption E. The authority needs to post decrypted tally $\sum_j v_j$ and a proof of correct decryption. Using the posted quantities on the board anyone can compute and verify tally to be valid, thus achieving *universal verifiability*.

Unlike hidden voter schemes where encoding of v_j was flexible, encoding of v_j in hidden vote schemes is limited. Since the encrypted votes are directly combined to compute the tally, the format of the votes has to be fixed. The encoding used is determined by the number of candidate choices (Cramer et al., 1997). Hence hidden vote schemes can only accommodate pre-specified candidate votes. Write-in votes are not handled.

Based on the public key used for the probabilistic, homomorphic encryption of votes we can further classify hidden vote schemes as: *vote threshold schemes*, *authority key threshold schemes*, and *voter key threshold schemes*.

6.1. *Vote threshold schemes*

In this category of hidden vote schemes, the vote is segmented into k shares by the voter using (t, k) secret sharing scheme, and each of the k authorities receives one encrypted share (encrypted with that authority's public key). Note that it can be the case that $t = k$. Each authority then uses homomorphic property of its public key cryptosystem and multiplies all the shares it received from voters to get the encrypted partial sum. Each authority then decrypts its partial sum and finally the authorities add their partial sums to get the final tally of votes. Schemes proposed in Cohen and Yung (1986) and Benaloh (1987) initiated this particular category. We describe them in detail below.

6.1.1. *Schemes in Cohen and Fischer (1985), Cohen and Yung (1986) and Benaloh (1987)*

These schemes introduced the seminal idea of using a probabilistic encryption scheme, with additive homomorphism, based on rth residuosity assumption (hardness of computing rth roots) (Stinson, 2002), for voting. We present the main idea here.

Given large primes p, q with $n = pq$ and a prime r such that $r|(p-1)$, the public key K of the authority is chosen such that $\gcd(K, n) = 1$ and that K is *not* a rth-residue mod n i.e. $K \neq x^r \ \forall x \in Z_n^*$. Only the encryption scheme E and (K, r, n) are made public to the voters. A voter V_j encrypts vote as:

$$E_K(v_j, u_j) = K^{v_j} u_j^r \pmod{n}, \tag{7}$$

where the vote $v_j \in \{0, 1\}$ and $u_j \in_R Z_n^*$ is a random string chosen by the voter (voter's receipt). During the pre-voting stage, V_j implements an interactive proof of validity of vote by posting encryptions of both votes (0, 1) and engaging the authority to complete a proof$_j$ (zero knowledge). During the voting stage, V_j designates one of the two encryptions as its desired vote. The tallying follows as described above.

6.1.2. Analysis of scheme in Benaloh (1987)

Scheme in Cohen and Fischer (1985) has only one authority and was modified in Cohen and Yung (1986), by making the voter share vote among k authorities (each authority A_i has public key K_i). Thus robustness to $k-1$ corrupt authorities is achieved. However, both these approaches are not robust to faults, since a non-participating authority can disrupt the voting. Hence the scheme in Benaloh (1987) was proposed, which used threshold secret sharing technique to ensure robustness to faults. The vote v_j is shared among k authorities using a (t, k) threshold scheme.

A *practical* limitation of Benaloh (1987) scheme is that the voting stage of the scheme is dependent on the termination of the pre-voting stage. To cast a vote, V_j has to wait for all other voters to finish their interactive proof phase. A variant scheme proposed in Iverson (1992) addresses this weakness and makes the V_j's participation independent of other voters. It achieves this by using the blind signature technique of Chaum et al. (1990). A blindly signed token containing voter ID is sent with the shares to the authorities. An invalid re-use of the token reveals the ID of the voter. However, this scheme lacks efficiency and robustness. Also, universal verifiability is sacrificed since the scheme does not use a bulletin board for communication.

Another disadvantage of Benaloh (1987) scheme is the use of interactive proofs that require intensive computations and communications. This weakness is eliminated by variants proposed in Sako and Killian (1994) and Cramer et al. (1996). The efficiency is improved in Sako and Killian (1994) by using different computational difficulty assumption (discrete log assumption instead of rth residuosity assumption) and making use of efficient non-interactive proofs. A similar but more robust approach (using threshold scheme) to achieve efficiency, is proposed in Cramer et al. (1996).

Though having some desirable properties, all the above schemes still require multiple computation and communication and hence are not *scalable* or *practical* in their current form. The scheme in Cramer et al. (1997), presented next, addresses this weakness and proposes an efficient category of schemes that require voter to perform only a single encryption for voting.

6.2. Authority key threshold schemes

Here the voter encrypts the vote with public key K of a tallying authority. To ensure robustness there are multiple authorities sharing the private (decryption) key among themselves using (t, k) *verifiable secret sharing scheme* (Pedersen, 1991), which also requires the use of threshold variants of public key cryptosystems. Such a scheme is proposed in Cramer et al. (1997) and is described below.

6.2.1. Scheme in Cramer et al. (1997)

This scheme achieves high efficiency by using a modified ElGamal encryption technique for encrypting the vote, based on the discrete log assumption (hardness of computing discrete log in multiplicative subgroups of Z_p^*, p being a large prime) (Stinson, 2002).

Announcement stage: The authorities engage in a shared-key generation protocol to share the private key S. The authorities publish the group public key $K = g^S$ and p, q (large primes). Authorities also publish g, h that are primitive elements (generators) of the unique multiplicative subgroup $G \subset Z_p^*$ of order q (p, q large primes, $q|(p-1)$).

Pre-voting, voting stages: The V_j computes encryption of vote $v_j \in \{-1, 1\}$ as:

$$E_K(v_j, r_j) = (g^{r_j}, K^{r_j} h^{v_j}) \pmod{p}, \tag{8}$$

where $r_j \in_R Z_q$ is the random string selected by the voter. Voter also computes a non-interactive proof of validity, proof$_j$, with zero-knowledge property.

Tallying stage: After checking each voter's proof of validity, t honest authorities combine together all the valid votes as:

$$\left(\prod_j g^{r_j}, \prod_j K^{r_j} h^{v_j}\right) = \left(g^{\sum_j r_j}, K^{\sum_j r_j} h^{\sum_j v_j}\right). \tag{9}$$

Using non-interactive proofs, the t authorities verifiably decrypt the product (without disclosing the private key S) to get the *exponentiated* tally of votes, $h^{\sum_j v_j}$. Next, the authorities search the tally space for a match. Assuming there are n voters who voted, the tally space is $\{h^{-n}, \ldots, 0, \ldots, h^n\}$. Hence, the final tally $\sum_j v_j$ is computed.

6.2.2. Analysis of the scheme in Cramer et al. (1997)

As seen from Tables 2 and 3, the scheme satisfies desirable properties and also achieves efficiency. Robustness to corrupt authorities is $t-1$ and robustness to faults is $k-t$. However, the vote format limits the scalability and practicality gain of this scheme. An increase in the number of candidates makes the proofs and the tallying process complex. For a simple election involving only two candidates and N voters, the tally computation complexity is $\Omega(N^{1/2})$. For elections involving M candidates, the complexity is exponential ($\Omega(N^{M-1/2})$). Scalability of tallying is improved significantly by Damgård and Jurik (2001), using generalized Paillier cryptosystem (based on composite residuosity assumption (Paillier, 1999)) instead of the ElGamal cryptosystem. We describe it below.

Given $n = pq$ where p, q are large primes, let $K \in Z_{n^2}$ with order of a non-zero multiple of n. (K, n) is public key of the Paillier cryptosystem. Encryption of a vote v_j is computed as:

$$E_K(v_j, u_j) = K^{v_j} r_j^n \pmod{n^2}, \tag{10}$$

where the vote $v_j \in Z_n$ and $r_j \in_R Z_n^*$. The decryption (and hence tallying) can be computed efficiently in this type of cryptosystem, instead of the brute-force search of the tally space done in Cramer et al. (1997).

An interesting scheme using Paillier's cryptosystem, was proposed by Baudron et al. (2001). The scheme addresses the hierarchical architecture of real-world elections. In a three-level hierarchy, voter post three encryptions, each with a different public key belonging to different set of authorities (precinct, regional and national). Tallying can be done reliably and accurately from the precinct level to national level. However, the complexity of proofs increases with the complexity of vote formats.

The receipt-freeness property can be incorporated using the techniques proposed in Hirt and Sako (2000) and Lee and Kim (2002). This is achieved by *not* allowing V_j to generate r_j. The approach of Hirt and Sako (2000) is similar to Sako and Killian (1995) but more robust, where k authorities (threshold sharing of the private key S) jointly generate the random string r_j using untappable channel from authority to voter, and non-interactive proofs (Jakobsson et al., 1996). However, as in Sako and Killian (1995), such an approach does sacrifice practicality and scalability properties for receipt-freeness. In Lee and Kim (2002), a *tamper-resistant randomizer* is used for randomizing the vote of the voter, along with non-interactive proofs. Since such a randomizer is local to the voting terminal, there is no need for untappable channel assumption. However, practicality of such a randomizer is a matter of concern. Another receipt-free approach, proposed in Baudron et al. (2001), is to employ a trusted third party randomizer with an untappable channel, which is more scalable (but less robust) than the technique of Hirt and Sako (2000).

6.3. Voter key threshold schemes

Though the above categories of hidden vote schemes can satisfy a number of properties, none of them so far seem to be *dispute-free*. This is mainly because of the *verifiable* shared-key generation protocol in the announcement stage which involves a number of interactions between the authorities and/or voters, that can lead to disputes. Voter key threshold schemes achieve dispute-freeness property. The voters act as the authorities, and participate to jointly share/generate their private keys, which are then used to encrypt their votes. The tally is computable as long as a threshold number of voters participate in voting and tallying. Such schemes are suitable for small scale elections where the presence of a separate authority is considered unnecessary as noted in Schoenmakers (1999). We present the main idea used in Schoenmakers (1999) below.

6.3.1. Scheme in Schoenmakers (1999)

The V_j uses a *publicly verifiable* (t, n) *secret sharing scheme*, to share a random secret s_j among the n voters (including self). V_j then posts its vote as:

$$E(v_j, s_j) = g^{v_j} g^{s_j} = g^{v_j + s_j}. \tag{11}$$

t honest voters can together compute $g^{\sum_j s_j}$ and then the tally $\sum_j v_j$ from $g^{\sum_j v_j + \sum_j s_j}$.

6.3.2. Analysis of the scheme in Schoenmakers (1999)

Since all the stages of this scheme are publicly verifiable, disputes can be resolved by anyone. *Scalability*, *efficiency* and *practicality* are traded for dispute-freeness property. The scheme also requires all voters to participate in the tallying stage. A variant of Schoenmakers (1999) proposed in Kiayias and Yung (2002) overcomes this particular weakness by satisfying a property defined as *self-tally* i.e. the tally is computable only when the last vote is cast. In addition to dispute-freeness, the scheme achieves the property of *maximal privacy* by simply making the threshold $t = n$. Each voter generates n random shares of a 0 (zero sharing) and distributes to n voters (including self). The private key s_j of V_j then becomes the sum of shares received from n voters (including self). When all the shares are combined by the voters, $\sum_j s_j = 0$, and hence, $g^{\sum_j v_j + \sum_j s_j} = g^{\sum_j v_j}$. In order to achieve *fairness*, the last voter is required to cast a mandatory zero vote, else the election result can be biased by the last voter. However, robustness property is lost since if one of the n voters does not send vote, then the election is disrupted and reactive measures will need to be implemented to compute the tally.

Due to the fixed format for votes, practical elections that include write-in votes cannot be implemented with hidden vote schemes using homomorphic encryption. Also the proof generation and verification, as well as vote encoding become complex and inefficient with the increase in the number of candidates, limiting the scope of this approach.

7. Hidden voter with hidden vote

To address the efficiency and vote format problems of *hidden vote* schemes, this third paradigm of voting schemes, was initiated by Fujioka et al. (1993) and Park et al. (1994). In the *hidden voter* class, there were no stringent limitations on the format of the vote and the tallying process was simple. However, the inherent fairness problem was its major disadvantage. Inaccuracy when detected made a re-election inevitable and since the partial tally was revealed the re-election was unfair. A solution to this problem is possible if the votes are also encrypted (hidden vote) instead of being open. This is the *hidden voter with hidden vote approach (HVHV)*.

In this approach, the voter uses the anonymous channel to send an *encrypted* vote to the tallying authority at the receiving end of the channel. We can think of HVHV class of schemes as a hybrid of the two previous classes. Based on the techniques used, we can further classify HVHV schemes into: *token based schemes*, *homomorphic encryption based schemes*, and *token and homomorphic encryption based schemes*.

7.1. Token based schemes

These schemes were derived from the hidden voter class. During registration the voter obtains a blind signature on an encryption of its vote from a registration authority. As in Fig. 6, the voter sends the signed hidden vote anonymously to a tallying authority as:

$$E_{K_m}\left(\text{sign}_A\left(E_K(v_j, r_j), K_A^{-1}\right)\right), \tag{12}$$

where E_{K_m} is the public key (K_m) encryption requirement of the anonymous channel (mixnet), and K is the public key of

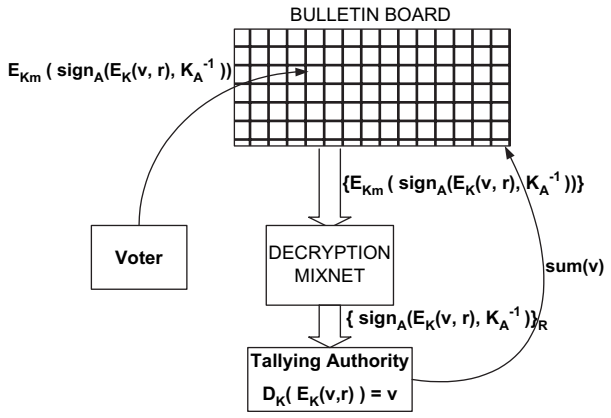

Fig. 6 – A typical token based HIDDEN VOTER with HIDDEN VOTE scheme, using blind signatures. During registration, voter obtains blind signature of an authority (with private key K_A^{-1}) on vote. Voter then posts signed encrypted vote, $E_{K_m}(\text{sign}_A(E_K(v, r), K_A^{-1}))$, on the bulletin board. K_m is the public key of the decryption mixnet. After all votes have been cast, the decryption mixnet is used to submit encrypted votes to tallying authority. During the tally stage, the tallying authority obtains K^{-1}, decrypts the votes and computes the tally $\text{sum}(v)$.

a tallying authority(s) or the voter, and K_A^{-1} is the private key of registration authority. v_j can be a pre-specified or un-specified vote. The tallying authority(s) share (or receive) the key K^{-1} to decrypt the votes and compute the tally. The scheme proposed in Fujioka et al. (1993) initiated this category and is described below.

7.1.1. *Scheme in Fujioka et al. (1993)*

The main idea used is to make $K = K_{V_j}$ in Eq. (12), where $K_{V_j}^{-1}$ is the private key of eligible V_j. After all signed tokens are received anonymously by the tallying authority, an *indexed list* of tokens is published. In order to decrypt the votes each V_j checks the indexed list for its token, and then sends the $(\text{index}, K_{V_j}^{-1})$ over an anonymous channel to the tallying authority. Hence the votes are decrypted and tally is then computed and posted on the bulletin board.

7.1.2. *Analysis of the scheme in Fujioka et al. (1993)*

Maximal fairness property is achieved, since even if *all* authorities collude, they would not be able to compute the partial tally. However, to achieve this the voter is required to participate in the tallying stage (*post-vote-casting*) which makes the scheme *impractical*. The main weakness of this scheme is however, the *accuracy* and *robustness*. Any abstainee can be detected by registration authority, and the authority could add votes for them. Collusion between voter and registration authority, and token collisions can also create inaccuracies.

The scheme in Baraani-Dastjerdi et al. (1995) tries to solve the robustness and accuracy by using multiple tallying authorities, and a trusted authority with untappable channels. But the approach is not really practical. In Juang et al. (2002) a more practical, scalable solution to the problem of accuracy and robustness, including the token collision problem was proposed. The main difference here is that the token contains an election tag, ID and signature of V_j (which solves collision problem), and the vote is encrypted separate from the token, using a public key K. The scheme has a set of registration authorities, using a threshold blind signature scheme (to avoid collusion between voter and registration authority). A set of k tallying authorities shares the private key K^{-1}. Here *maximal fairness* property is traded for accuracy and practicality, with fairness only up to $k-1$ internal adversaries.

The receipt-freeness weakness of Fujioka et al. (1993) was addressed by Okamoto (1997). The main idea of the scheme is to use trapdoor bit commitment technique as:

$$E_{K_{V_j}}(v_j, r_j) = \text{comm}\left(v_j, r_j, K_{V_j}\right) = g^{v_j} g^{s_j r_j} = g^{v_j + s_j r_j}, \tag{13}$$

where $g \in G$, $s_j \in Z_q$ is the secret of the voter with $K_{V_j} = g^{s_j}$. The random string $r_j \in Z_q$ is the trapdoor that allows the voter to construct:

$$\text{comm}\left(v_j', r_j', K_{V_j}\right) = \text{comm}\left(v_j, r_j, K_{V_j}\right). \tag{14}$$

This enables the voter V_j to prove the vote to be v_j' or v_j. Hence the scheme achieves receipt-freeness property. However, Okamoto (1997) assumes availability of *anonymous untappable channels* (for sending v_j, r_j) which is not a practical assumption.

7.2. *Homomorphic encryption based schemes*

The above category based on tokens, could not satisfy accuracy as well as universal verifiability. Hidden vote schemes based on homomorphic encryption were able to satisfy these properties. Hence a second category of HVHV schemes, based on homomorphic encryption technique was proposed. Such a scheme appeared in Park et al. (1994) and was subsequently improved in verifiability and robustness (Ogata et al., 1997; Jakobsson, 1998; Abe, 2000; Golle et al., 2002). The general structure of these schemes in stages is as follows.

Announcement stage: Authorities publish the public key $K = g^S$ and $g \in G$ as seen before. $S \in Z_q$ is shared by a group of tallying authorities using a (t, k) threshold scheme.
Pre-voting stage: n voters register, and each registered V_j then computes:

$$E_K(v_j, r_j) = \left(g^{r_j}, K^{r_j} v_j\right) = (X, Y), \quad r_j \in_R Z_q, \tag{15}$$

where $v_j \in G$ can be a pre-specified or an un-specified vote.
Voting stage:

(1) *Vote casting*: As shown in Fig. 7, V_j posts encrypted vote $E_K(v_j, r)$ in designated section on the board. An optional check might be done by authority (depending on the mixnet employed) just to ensure format of the n encrypted votes.
(2) *Mixing*: The batch of n encrypted votes is sent through the verifiable, re-encryption mixnet, containing m mixes. The output of the mixnet will contain a batch of mixed encrypted votes, each vote of the form:

$$E_K\left(v_{\pi(j)}, r_{\pi(j)} + R\right) = \left(g^{r_{\pi(j)}}, K^{r_{\pi(j)}+R} v_{\pi(j)}\right) = (X', Y'), \tag{16}$$

where $\pi : n \to n$ denotes the random permutations of the m mixes, and R denotes the sum of random numbers used by

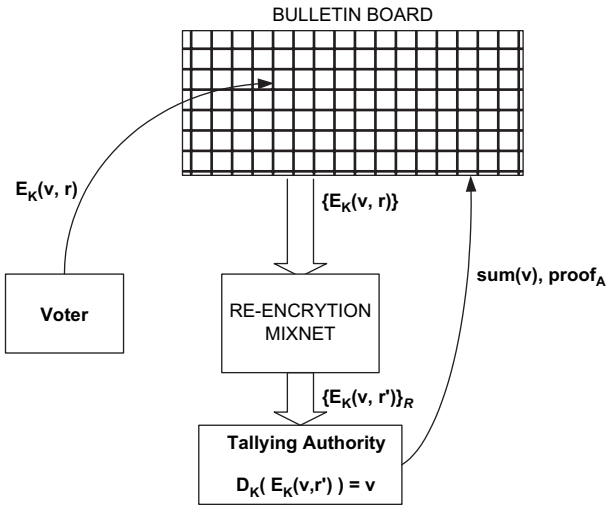

Fig. 7 – A typical homomorphic encryption based HIDDEN VOTER with HIDDEN VOTE scheme. Voter submits encrypted vote $E_K(v, r)$, to its authenticated section in the bulletin board. After all votes have been cast, the re-encryption mixnet is used to submit re-encrypted and permuted votes to tallying authority. The votes are decrypted to get the tally $\mathrm{sum}(v)$ and is posted along with proof of correct decryption proof_A.

the mixes to re-encrypt.[3] Proofs are generated from the mixing process. Optionally, these proofs may be posted on the bulletin board. The mixing process is verified to be correct.

Tallying stage: Using the private key S, verifiable decryption of each vote is performed as:

$$v_{\pi(j)} = Y'/(X')^S. \quad (17)$$

A non-interactive proof of correct decryption, proof_A, is posted. The votes are verified to be valid and are then counted to generate the final tally.

Verification stage: Depending on the mixnet used, anyone can verify (conditionally (Golle et al., 2002) or universally verifiable (Neff, 2001)) that the tally is accurate.

7.2.1. Analysis of homomorphic based HVHV schemes

The mixnet being the main protocol in these schemes, determines the properties satisfied. If a universally verifiable, robust re-encryption mixnet (such as Neff, 2001; Furukawa and Sako, 2001) is used, then properties including eligibility, privacy, accuracy, fairness, and robustness can be satisfied. The main properties that separate the HVHV schemes within this category, are the scalability (efficiency) and robustness of the mixnet. Most of the schemes can be modified to satisfy receipt-freeness property. Such a modification is proposed in Lee et al. (2003), where the idea (as seen in Section Authority key threshold schemes) is to use a tamper-resistant randomizer that generates the random string r_j for the voter V_j using non-interactive proof (Jakobsson et al., 1996).

[3] R is different for each element at the output of the mix. We avoid complexity of notation.

However, this receipt-free approach is not generally applicable to all schemes. This is due to the fact that some re-encryption mixnets, such as that of Golle et al. (2002), require V_j to prove knowledge of r_j used. This step is taken to address *malleability* of ElGamal encryption i.e. a vote of the form $E_K(v_j, r_j)$ can be duplicated by another voter V_l as $E_K(v_j, r_j + r_l)$. By doing this, a *coerced* V_l can affect the tally randomly or try to gain statistical information about V_j's choice (corrupt V_l). The approach of Lee et al. (2003) hides r_j from V_j and the proof of knowledge of r_j cannot be provided by V_j.

As a variant of the above category, an HVHV scheme that extends a hidden vote scheme to include write-in votes is proposed in Kiayias and Yung (2004). The main idea is to employ a hybrid approach of using a re-encryption mixnet (for write-in votes) and a homomorphic encryption scheme (for regular, pre-specified votes). The voter generates a *vector* vote that contains three fields, one for a pre-specified vote (among a set of *c* candidates), a flag bit and a write-in vote field. Initially, all fields are set to 0. If the flag bit is 1 it indicates that there is a write-in vote (for an unspecified candidate). The scheme *limits* the option of voting for either a pre-determined candidate or a write-in vote, but not for both. The voter posts the encrypted vector vote along with a proof of validity of vote. The first field of all vector votes is combined and decrypted to obtain the tally for the pre-specified candidates. The write-in votes are handled separately, in batches of certain size, by sending them through a re-encryption mixnet. To ensure privacy of voter, a heuristic measure of adding random number of blank write-in votes in each batch is proposed. The output of the mixnet is then used to tally the write-in votes. However, the scheme sacrifices efficiency and scalability as the number of write-in votes increases. Also, like other HVHV schemes above, the incoercibility property is not satisfied.

7.3. Token and homomorphic encryption based schemes

This category of HVHV schemes, try to satisfy receipt-freeness and incoercibility properties. Schemes proposed in Juels and Jakobsson (2002) and Acquisti (2004) belong to this category and their main idea is presented below.

7.3.1. Schemes in Juels and Jakobsson (2002) and Acquisti (2004)

During registration stage, voter obtains a unique token encrypted with a public key that is shared by *k* authorities. The public key encryption used is homomorphic. Voter sends encrypted vote combined with the encrypted token, over an *anonymous broadcast channel* to a bulletin board with *no* designated sections. Each voter's token allows multiple votes to be cast, but only one encrypted vote per token is considered for decryption. The authorities *blindly* compare each of the submitted encryptions with a mixed (using re-encryption mixnet) set of valid encrypted tokens, to determine the validity of the voter and the associated vote before decrypting the vote. The approach of Acquisti (2004) improves on Juels and Jakobsson (2002) by accommodating write-in votes.

7.3.2. Analysis of schemes in Juels and Jakobsson (2002) and Acquisti (2004)

Receipt-freeness property is achieved by allowing the voter to prove encrypted token/vote in more than one way. This also

enables the voter to give fake token to adversary. Since adversary cannot confirm the token nor the abstention of voter, and since the voter can vote multiple times without being traced by adversary, the incoercibility property is indeed satisfied, but conditionally. This is because the schemes assume that adversary is not present in the registration stage, hence not addressing the presence of adversary during all stages of the scheme. Unfortunately, scalability, universal verifiability and accuracy properties are traded for incoercibility. The anonymous broadcast channel with no designated section on the bulletin board, can also be difficult to implement.

8. Discussion

Having looked at the various schemes under our classification, we now contrast the approaches. Tables 2 and 3 provide a comparison of voting schemes that we have discussed in the previous sections. A graphical illustration of the design space for generic voting schemes and the three classes of voting schemes is shown in Figs. 8 and 9 .

8.1. *Comparison between the classes*

(1) *Hidden voter schemes:* The main advantages of hidden voter schemes, are that tallying process is the simplest among all the three classes, and computation at the voter end is also simple. However, in these schemes accuracy, fairness and robustness cannot be satisfied together. Inaccuracies in the tally can only be resolved by another election which is in no way fair. Moreover, the voter participation requirement can be heavy. Anonymous channel implementation with robust, verifiable decryption mix-nets can reduce the voter participation, but such channels are still cumbersome and not really efficient when used for large scale elections.

(2) *Hidden vote schemes:* The voter participation is minimal and universal verifiability property is easy to achieve. More importantly, there is no requirement for any form of mix-nets. Small scale elections (e.g. boardroom elections) can benefit from simplicity of hidden vote schemes since they can be designed to work without any authority, using the dispute-freeness property. The biggest drawback with hidden vote schemes is that vote format is not flexible to accommodate write-in votes. Also some vote formats proposed involve complex computations for the voter and at times for tallying. However, complexity for simple 1-of-2 candidates election can be relatively efficient compared to HVHV approaches.

(3) *Hidden voter with hidden vote schemes:* The main advantages of HVHV are the flexible vote format (including write-in) and relatively low voter computation (no complex proofs are usually necessary), which are desirable properties for large scale elections. However, anonymous channel implementation is an issue in HVHV schemes as in hidden voter schemes, although to a lesser extent (see below). Trading scalability of scheme for achieving universal verifiability and accuracy using the mixnet, is a factor in deciding between hidden vote and the HVHV approach. The tallying process itself can be tedious since it requires individual encrypted vote validation, decryption, and vote validation followed by the actual tallying. Hence, while in hidden vote the post-vote-casting process involves the verification of proofs and tallying, in HVHV the post-vote-casting process involves time consuming mixing and tallying.

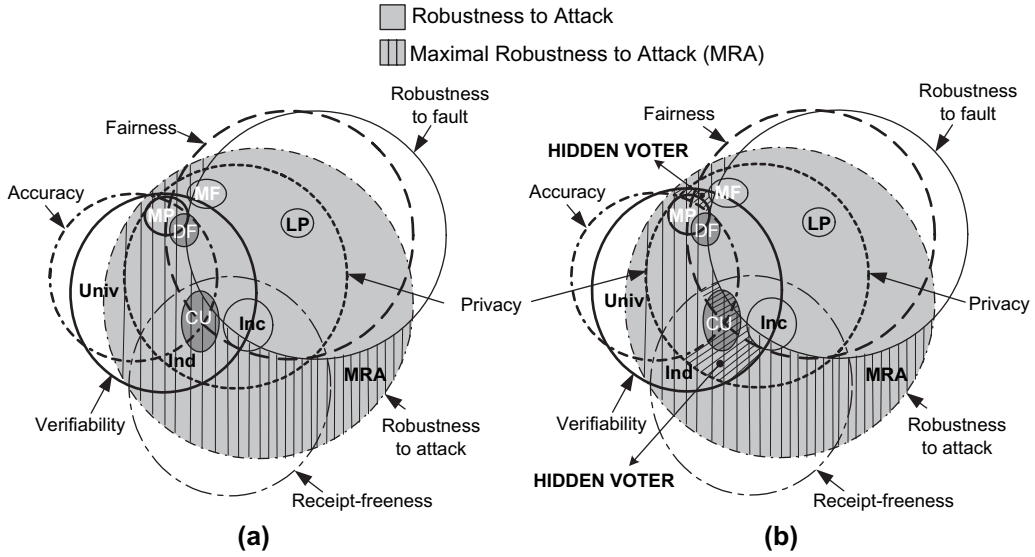

Fig. 8 – Notation: DF – dispute freeness; Univ – universally verifiable; Ind – individually verifiable; Inc – incoercibility; MP – maximal privacy; MF – maximal fairness; LP – long-term privacy; CU – conditionally universally verifiable. The intersection of verifiability with accuracy defines Universal Verifiability (UV) and the non-intersection includes only Individual Verifiability (Ind). (a) Design space available for voting schemes. As seen, various tradeoffs exist in the design of a voting scheme. The overlapping regions are the design spaces. Selection of any design space leads to satisfication of certain properties at the expense of sacrificing one or more of the remaining properties. (b) Design space of hidden voter schemes. The hashed area indicated is the design space.

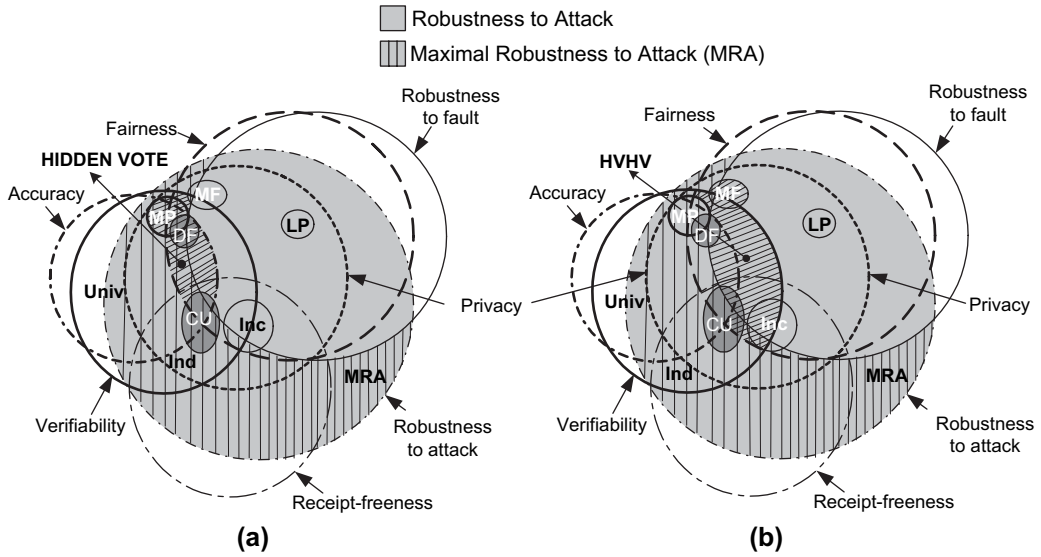

Fig. 9 – Notation: DF – dispute freeness; Univ – universally verifiable; Ind – individually verifiable; Inc – incoercibility; MP – maximal privacy; MF – maximal fairness; LP – long-term privacy; CU – conditionally universally verifiable. The hashed areas indicated are the design spaces for voting schemes. (a) Design space of hidden vote schemes. (b) Design space of hidden voter with hidden vote schemes.

8.2. *Comparison between decryption mixnets and re-encryption mixnets*

The promising HVHV schemes are based on re-encryption mixnets while the hidden voter schemes are based on the decryption mixnets. Both forms of mixnets have problems, with robustness and verifiability being a deterrent to scalability. The re-encryption mixnets however, are inherently more robust to faults than decryption mixnets. This is essentially due to the fact that in re-encryption mixnet, the key used to encrypt the votes is separate from the mixes, which just re-encrypt and permute the input batches. But in the case of decryption mixnets, the public key of the mixnet is used to encrypt the votes. Hence, in the case of re-encryption mixnets, the problem of robustness is only limited to ensuring correct mixing by the mixes, in decryption mixnets (in addition to ensuring correctness) the public key of the faulty mix will still need to be decrypted to compute the votes. Finally, to the best of our knowledge, while re-encryption mixnets with universal verifiability property have been proposed (Furukawa and Sako, 2001; Neff, 2001), construction of robust decryption mixnets satisfying universal verifiability remains an open problem.

8.3. *Tradeoffs*

Figs. 8 and 9 indicate that common security property tradeoffs exist in the three classes of voting schemes. An analysis of some important tradeoffs is presented below.

Maximal privacy property which allows the privacy of a voter to be breached *only* with a collusion of all remaining entities (voters and authorities), while desirable, requires all the voters to either participate in the post-vote-casting stage or to mandatorily cast their votes (i.e. no abstaining). Hence maximal privacy is often traded with fairness, robustness, scalability and practicality. *Fairness* property has the same notion of privacy preservation against corrupt authorities. Achieving *maximal fairness* property, which prevents computation of partial tally even with a collusion of *all* authorities, requires the voter to participate in the post-vote-tallying stage. Hence maximal fairness property conflicts with practicality property.

Long-term privacy property was not discussed with any of the schemes so far. To satisfy this property, all the communications between entities in the voting scheme, require a private channel using private keys. For distribution of the private keys to the entities, we further require untappable channels. These requirements are hard to realize on a large scale, hence essentially long-term privacy is traded by voting schemes for the scalability and practicality properties.

Receipt-freeness property does not imply *incoercibility*. Schemes that satisfy receipt-freeness, assume absence of adversary when vote is cast, and hence do not address incoercibility property. Incoercible schemes are receipt-free, since the voter is allowed to vote multiple times and *also* able to prove its vote in more than one way. Moreover, the link between the voter and its encrypted vote has to be eliminated to ensure that the multiple voting is not traceable to the voter. Due to this reason *hidden vote, and hidden voter schemes cannot satisfy incoercibility*. Achieving incoercibility also leads to sacrificing universal verifiability and hence accuracy. We note that the design of a general receipt-free approach for the HVHV schemes using re-encryption mixnets, is also an open problem. Finally, in the HVHV schemes that address incoercibility, an assumption that adversary is not representing the voter during registration becomes necessary. Under this assumption a scheme satisfies incoercibility only conditionally.

9. Conclusion

In this paper, we have provided a framework under which performance of secure voting schemes can be evaluated and

compared. Voting schemes since the seminal work of Chaum (1981) and Cohen and Fischer (1985), lead to three distinct classes: hidden voter, hidden vote and the hidden voter with hidden vote (HVHV). After defining a set of security properties, we investigated each class chronologically, to identify the properties satisfied. On comparison under our framework, we found that there is no clear potential candidate. While hidden vote class does have many desirable properties including dispute-freeness, the vote format and incoercibility weaknesses limit its application to practical elections. On the other hand, HVHV class has more practical features including incoercibility, but is limited by the scalability of the re-encryption mixnet. We see that a hybrid of a hidden vote and a HVHV approach has appeal when implementing a scalable and practical voting scheme. Recent proposition by Kiayias and Yung (2004) takes an initial step in this direction.

Research in mixnets is a growing area with many applications. We presented two classes of mixnets and how they are integrated into voting schemes to achieve anonymity. In order to extend secure electronic voting to a public network, we also included incoercibility in our framework. A scalable, and accurate e-voting scheme satisfying incoercibility can be a potential candidate for voting over a public network such as the Internet. We believe that our framework will allow designers to check what conditions their schemes satisfy and what are the tradeoffs, thus avoiding making any unwarranted claims or unintended errors.

REFERENCES

Abe M. Universally verifiable mix-net with verification work independent of the number of mix-servers. IEICE Transactions on Fundamentals July 2000;E83-A(7). p. 1431–40.

Acquisti A. Receipt-free homomorphic elections and write-in ballots. Cryptology ePrint Archive, Report 2004/105, <http://eprint.iacr.org/>; 2004.

Baraani-Dastjerdi A, Pieprzyk J, Safavi-Naini R. A practical electronic voting protocol using threshold schemes. In: Proceedings of the 11th annual computer security applications conference 1995. p. 143–8.

Baudron O, Foque PA, Pointcheval D, Poupard G, Stern J. Practical multi-candidate election system. In: Proceedings of the 20th ACM symposium on principles of distributed computing. ACM Press; 2001. p. 274–83.

Benaloh J. Verifiable secret-ballot elections, Ph.D. thesis, Yale University; 1987.

Benaloh J, Tuinstra D. Receipt-free secret-ballot elections. In: Proceedings of the 26th ACM Symposium on the Theory of Computing (STOC). ACM; 1994. p. 544–53.

Blum M, De Santis A, Micali S, Persiano G. Non-interactive zero-knowledge. SIAM Journal of Computing 1991;20(6):1084–118.

Boyd C. A new multiple key cipher and an improved voting scheme. In: Advances in cryptology – EUROCRYPT '89. Springer-Verlag; 1990. p. 617–25.

Cal Tech-MIT. Voting: what is, what could be. Cal Tech-MIT Voting Technology Project Report, <www.vote.caltech.edu/Reports/>; 2001.

Chaum D. Untraceable electronic mail, return addresses, and digital pseudonyms. Communications of the ACM 1981;24(2):84–8.

Chaum D. Blind signature system. In: Advances in cryptology – CRYPTO '83. Plenum Press; 1984. p. 153.

Chaum D. Elections with unconditionally-secret ballots and disruption equivalent to breaking RSA. In: Advances in cryptology – EUROCRYPT '88. LNCS, vol. 330. Springer Verlag; 1988a. p. 177–82.

Chaum D. The dining cryptographers problem: unconditional sender and recipient untraceability. Journal of Cryptology 1988b;1(1):65–75.

Chaum D. Secret-ballot receipts: true voter-verifiable elections. IEEE Security & Privacy Magazine Feb 2004.

Chaum D, Fiat A, Naor M. Untraceable electronic cash [Extended Abstract]. In: Advances in cryptology – CRYPTO '88. LNCS, vol. 403. Springer-Verlag; 1990. p. 319–27.

Chor B, Goldwasser S, Micali S, Awerbuch B. Verifiable secret sharing and achieving simultaneous broadcast. In: Proceedings of IEEE foundation of computer science 1985. p. 335–44.

Cohen (Benaloh) JD, Fischer MJ. A robust and verifiable cryptographically secure election scheme. In: Proceedings of 26th symposium on foundation of computer science 1985. p. 372–82.

Cohen (Benaloh) JD, Yung M. Distributing the power of a Government to enhance the privacy of voters. In: ACM symposium on principles of distributed computing 1986. p. 52–62.

Cramer R, Franklin M, Schoenmakers B, Yung M. Multi-authority secret ballot elections with linear work. In: Advances in cryptology – EUROCRYPT '96. LNCS, vol. 1070. Springer Verlag; 1996. p. 72–83.

Cramer Ronald, Gennaro Rosario, Schoenmakers Berry. A secure and optimally efficient multi-authority election scheme. In: Advances in cryptology – EUROCRYPT '97. LNCS, vol. 1233. Springer-Verlag; 1997. p. 103–18.

Damgård I, Jurik M. A generalisation, a simplification and some applications of Paillier's probabilistic public-key system. In: Proceedings of public key cryptography, fourth international workshop on practice and theory in public key cryptography, PKC 2001, LNCS, vol. 1992. Springer-Verlag; 2002. p. 119–36.

ElGamal T. A public-key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory July 1985;31(4):469–72.

Fiat A, Shamir A. How to prove yourself: practical solutions to identification and signature problems. In: Advances in cryptology – CRYPTO '86. LNCS, vol. 263. Springer-Verlag; 1987. p. 186–94.

Fujioka A, Okamoto T, Ohta K. A practical secret voting scheme for large scale elections. In: Advances in cryptology – AUSCRYPT '92. LNCS, vol. 718. Springer-Verlag; 1993. p. 248–59.

Furukawa J, Sako K. An efficient scheme for proving a shuffle. In: Advances in cryptology – CRYPTO '01. LNCS, vol. 2139. Springer-Verlag; 2001. p. 368–87.

Goldwasser S, Micali S, Rackoff C. The knowledge complexity of interactive proof systems. SIAM Journal on Computing 1989;18:186–208.

Golle P, Juels A. Dining cryptographers revisited. In: . Advances in cryptology – EUROCRYPT '04, vol. 3027. Springer-Verlag; 2004. p. 456–73.

Golle P, Zhong S, Boneh D, Jakobsson M, Juels A. Optimistic mixing for exit-polls. In: ASIACRYPT '02. LNCS, vol. 2501. Springer-Verlag; 2002. p. 451–65.

Gritzalis D, editor. Secure electronic voting. Advances in information security, vol. 7. Kluwer Academic Publishers; 2002a.

Gritzalis D. Principles and requirements for a secure e-voting system. Computers & Security 2002b;21(6):539–56.

Groth J. A verifiable secret shuffle of homomorphic encryptions. In: Proceedings of public key cryptography, sixth international workshop on practice and theory in public key cryptosystems, PKC 2003. LNCS, vol. 2567; 2003. p. 145–60.

Hirt M, Sako K. Efficient receipt-free voting based on homomorphic encryption. In: Advances in cryptology – EUROCRYPT '00. LNCS, vol. 1807. Springer-Verlag; 2000. p. 539–56.

Iverson KR. A cryptographic scheme for computerized general elections. In: Advances in cryptology – CRYPTO '91. LNCS, vol. 576. Springer-Verlag; 1992. p. 405–19.
Jakobsson M. A practical mix. In: Advances in cryptology – EUROCRYPT '98. LNCS, vol. 1403. Springer-Verlag; 1998. p. 449–61.
Jakobsson M, Sako K, Impagliazzo R. Designated verifier proofs and their applications. In: Advances in cryptology – EUROCRYPT '96. LNCS, vol. 1070. Springer-Verlag; 1996. p. 143–54.
Jakobsson M, Juels A, Rivest R. Making mix nets robust for electronic voting by randomized partial checking. In: Proceedings of USENIX security '02 2002. p. 339–53.
Jefferson D, Rubin AD, Simons B, Wagner D. A security analysis of the secure electronic registration and voting experiment (SERVE). Technical Report, <http://servesecurityreport.org/>; 2004.
Juang W, Lei C, Liaw H. A verifiable multi-authority secret election allowing abstention from voting. The Computer Journal 2002; 45(6):672–82.
Juels A, Jakobsson M. Coercion-resistant electronic elections. Cryptology ePrint Archive, Report 2002/165, <http://eprint.iacr.org/>; 2002.
Kiayias Aggelos, Yung Moti. Self-tallying elections and perfect ballot secrecy. In: Proceedings of public key cryptography, fifth international workshop on practice and theory in public key cryptosystems, PKC 2002. LNCS, vol. 2274. Springer-Verlag; 2002. p. 141–58.
Kiayias Aggelos, Yung Moti. The vector-ballot e-voting approach. In: Financial cryptography. LNCS, vol. 3110. Springer-Verlag; 2004. p. 72–89.
Kohno T, Stubblefield A, Rubin AD, Wallach DS. Analysis of an electronic voting system. IEEE symposium on security and privacy; May 9–12, 2004.
Lee B, Kim K. Receipt-free electronic voting scheme with a tamper-resistant randomizer. In: ICISC '02. LNCS, vol. 2587. Springer-Verlag; 2002. p. 389–406.
Lee B, Boyd C, Dawson E, Kim K, Yang J, Yoo S. Providing receipt-freeness in mixnet-based voting protocols. In: Proceedings of the ICISC '03, 2003. p. 261–74.
Neff CA. A verifiable secret shuffle and its application to e-voting. In: Proceedings of the eighth ACM conference on computer and communications security. ACM Press; 2001. p. 116–25.
Ogata W, Kurosawa K, Sako K, Takatani K. Fault-tolerant anonymous channel. In: Proceedings of the ICICS '97. LNCS, vol. 1334; 1997. p. 440–4.
Okamoto T. Receipt-free electronic voting schemes for large scale elections. In: Proceedings of the workshop on security protocols '97. LNCS, vol. 1361. Springer-Verlag; 1997. p. 25–35.
Paillier P. Public-key cryptosystems based on composite degree residue classes. In: Advances in cryptology – EUROCRYPT '99. LNCS, vol. 1592; 1999. p. 223–38.
Park C, Itoh K, Kurosawa K. Efficient anonymous channel and all/nothing election scheme. In: Advances in cryptology – EUROCRYPT '93. LNCS, vol. 765. Springer Verlag; 1994. p. 248–59.
Pedersen T. A threshold cryptosystem without a trusted party. In: Advances in cryptology – EUROCRYPT '91. LNCS, vol. 547. Springer-Verlag; 1991. p. 522–6.
Pfitzmann B. Breaking an efficient anonymous channel. In: Advances in cryptology – EUROCRYPT '94. LNCS, vol. 950. Springer-Verlag; 1994. p. 332–40.
Rivest R, Shamir A, Adleman L. A method for obtaining digital signatures and public key cryptosystems. Communications of the ACM 1978;21:120–6.
Sako K, Killian J. Secure voting using partially compatible homomorphisms. In: Advances in cryptology – CRYPTO '94. LNCS, vol. 839. Springer-Verlag; 1994. p. 411–24.
Sako K, Killian J. Receipt-free mix-type voting scheme – a practical solution to the implementation of a voting booth. In: Advances in cryptology – EUROCRYPT '95. LNCS, vol. 921. Springer-Verlag; 1995. p. 393–403.
Schoenmakers B. A simple publicly verifiable secret sharing scheme and its applications to electronic voting. In: Advances in cryptology – CRYPTO '99. LNCS, vol. 1666. Springer-Verlag; 1999. p. 148–64.
Shamir A. How to share a secret. Communications of the ACM 1979;22(11):612–3.
Stinson DR. Cryptography: theory and practice. 2nd ed. CRC Press; 2002.

Krishna Sampigethaya is a PhD candidate in the Department of Electrical Engineering, and a graduate research assistant in the Network Security Lab at the University of Washington, Seattle, USA. He holds an MSEE in Communications and Networking, and is conducting research on protocols related to secure electronic voting, and on location privacy in vehicular networks.

Radha Poovendran is an assistant professor in the Department of Electrical Engineering, and the founding director of the Network Security Lab at the University of Washington, Seattle, USA. He received his Ph.D. in Electrical Engineering from the University of Maryland, College Park, USA in 1999. His research interests are in the areas of applied cryptography for multiuser environment, networking, and Information Theory. He is a recipient of NSA Rising Star, NSF CAREER, ARO YIP, ONR YIP, and PECASE awards for his research contributions in the areas of wired and wireless security. He is also a recipient of a number of awards for teaching and advising within the university community. He is a founding member and the Associate Director for Research of the UW Center for Information Assurance and Cyber security.