

An Introduction to Electronic Voting Schemes*

Andreu Riera

Unitat de Combinatòria i de Comunicació Digital
Universitat Autònoma de Barcelona

e-mail: ariera@ccd.uab.es

Abstract

The field of secure voting schemes has recently attracted much attention from researchers. Since many security requirements are involved (and some of them are contradictory) solutions are not obvious. The aim of this paper is to provide an introduction to the state-of-the-art in electronic voting schemes. The cryptographic techniques most commonly used to design secure elections will be presented and discussed.

Keywords: Network security, cryptographic protocols, electronic voting schemes

1 Introduction

The ultimate goal of a computer network is to move information around. By moving information, people need not move. Home banking, home working, or remote ticket selling are examples of applications that use computer networks in this way. Yet another example of application is an electronic voting scheme. The objective of electronic voting schemes is to allow elections to take place over general-purpose and open computer networks.

*This work has been partially funded by the Spanish Government Commission CICYT, through its grant TEL97-0663.

It is obvious that some security measures have to be included in the design of all mentioned examples of communication applications. The first step in this process should consist of listing all security requirements desired for the application. Threats and vulnerabilities of the system have to be identified. Next, the appropriate security mechanisms and techniques have to be chosen (and adequately used) in order to counteract all possible attacks. Sometimes (not uncommonly in complex systems such as electronic voting schemes), some of the security requirements need somehow contradictory solutions. This forces the designer to find a balance among the desired security properties of the application.

The first proposal of electronic voting scheme [Cha81] was included in a broader work regarding anonymous communication. Since then, many other concrete electronic voting schemes have been devised. They range from merely theoretical proposals to schemes which have been actually implemented over computer networks. Still, some of the commonly accepted security requirements do not have a satisfactory solution yet. Research is currently being undertaken on a number of open problems. In addition, electronic voting schemes have to face another important challenge: to gain social acceptance.

The aim of this report is to give a descriptive introduction to the state-of-the-art in electronic voting schemes. No concrete topics will be explained in detail because it is intended to be a general reading. The basic structure and security requirements of electronic voting schemes will be presented, together with the most common ways of achieving them. Therefore, some of the existent electronic voting schemes will be better covered than others.

Section 2 briefly describes the participants of an electronic election and the different voting phases. In Section 3, a list of security requirements is given. The discussion on how electronic voting schemes manage to fulfill the desired requirements will be done following a generic example. The example shown is not intended to be a precise proposal of voting scheme, but just a compilation of the main steps and actions that usually take place in most proposals. In order to allow a clear presentation of the example, some of the specific cryptographic techniques used will be reviewed in Section 4. The generic electronic voting scheme is then presented in Section 5. Section 6 is devoted to the discussion on how the used security mechanisms and tools assure the desired requirements. Finally, Section 7 introduces practical issues regarding implementation.

2 Participants and voting phases

Broadly speaking, the participants involved in an electronic election are a collective of voters and a set of voting authorities. The number and purpose of the different voting authorities vary, depending on the precise electronic voting scheme considered. Simpler systems consist of a single voting authority which includes all voting functionality, while more complex systems divide the voting functionality among several voting authorities, each with a well defined task. There are even some voting schemes which do not need any voting authority at all, and others that need the involvement of external participants. As it can be seen, there is a wide range of possibilities. However, only those schemes that consider the existence of, at least, one voting authority, are suitable to be operated realistically over a computer network. Voting schemes with no need for voting authorities are based on some kind of secure multiparty computation among the set of voters. As a consequence, such schemes cannot be implemented practically for more than a handful of people, because computation and communication costs are prohibitive. Therefore, they only have theoretical interest.

Regarding voting phases, an electronic election can usually be divided into the following phases:

- **Registration phase**

During this first stage, a certain voting authority creates the electoral roll and publishes it on the network. During a certain complaining period, voters should be able to post objections. After it, the final version of the electoral roll is published by the voting authority.

- **Voting phase**

During the voting phase, voters cast the desired ballots using the communication facilities of the network. The vote casting procedure may need a single session with a ballot collecting authority (which can be the same authority as the one responsible for the electoral roll). Some voting schemes need more than a single session, and it is not uncommon to have more than one voting authority involved in the voting phase.

- **Counting phase**

At the end of the voting phase, the ballot collecting authority stops accepting ballots. The counting process is initiated. Finally, the tally

is published and made available through the network.

3 Security requirements

An electronic voting scheme is a complex system, facing a number of compulsory security requirements. A single cryptographic protocol or mechanism alone is not enough to assure all desired properties. This is the reason for talking about “voting schemes” and not just “voting protocols”. A number of cryptographic tools and protocols must be used together to guarantee the desired security features. Some different lists of security requirements for electronic voting schemes have been proposed. Nonetheless, the level of agreement normally turns out to be high. The following list of four essential security requirements comes by taking the list of properties proposed in [CC96] as a basis, and adding the third privacy’s factor:

- **Accuracy:** A voting scheme is accurate if (1) it is not possible for a validated ballot to be altered, (2) it is not possible for a validated ballot to be eliminated from the final tally, and (3) it is not possible for an invalid ballot to be counted in the final tally.
- **Democracy:** A voting scheme is democratic if (1) it permits only eligible voters to vote, and (2) it ensures that each eligible voter can vote only once.
- **Privacy:** A voting scheme is private if (1) neither election authorities nor anyone else can link any ballot to the voter who has cast it, (2) no voter can prove that he voted in a particular way, and (3) all ballots remain secret while the voting is not completed.
- **Verifiability:** A scheme is verifiable if voters can independently verify that their ballots have been counted correctly.

4 Related cryptographic techniques

4.1 Authentication and authenticated key exchanges

In most of the communication sessions involved in any secure application over a computer network, it is needed to have assurance of the identity of

the party at the other side of the wire. There are many ways to reach this assurance. For example, classical passwords serve this purpose. However, in open networks the most secure and convenient way (for many reasons) is known as strong authentication, and it involves public key cryptography. Using strong authentication protocols, any individual has to be previously bound to a certain public key. The most widely accepted way of establishing this binding is by certificates and certification authorities [CCI88b]. Authentication procedures consist of, in some way, proving knowledge of the corresponding private key.

Using a strong authentication protocol, two communicating parties can verify the identity of each other. This is called peer-to-peer (bilateral) authentication. However, if authentication alone is used, nothing prevents an active attacker settled between two legitimate parties to impersonate one of them *after* authentication has taken place. This problem is counteracted through an authenticated key exchange during or following the authentication phase. By carrying out an authenticated key exchange, both legitimate parties exchange a secret piece of information that will serve as a symmetric session key. Authentication is linked to the key exchange, to have confidence of whom each is sharing the secret with. The exchanged session key is used to encrypt all subsequent transmitted user data, thus providing message confidentiality, authenticity and integrity. A very good discussion on authenticated key exchanges can be found in [DOW92].

4.2 Blind signatures

Blind signature mechanisms, first introduced by Chaum [Cha83, Cha85], allow some party to get a message digitally signed by another party, without revealing any information about the message to the signer.

Chaum demonstrated the implementation of this concept using RSA signatures [RSA78] as follows: Suppose Alice has a message m (actually m would be the digest of the real message to be signed) that she wishes to have signed by Bob, and she does not want Bob to learn anything about m . Let (n, e) be Bob's RSA public key and (n, d) be his private key. Alice generates a random value r (called a blinding factor), such that $\gcd(r, n) = 1$. Alice sends to Bob

$$m' = r^e \cdot m \bmod n$$

Since the value m' is “blinded” by the random value r , Bob cannot derive useful information from it. Bob returns the signed value

$$s' = (m')^d = (r^e \cdot m)^d \bmod n$$

to Alice. Note that $s' = r \cdot m^d \bmod n$. Therefore, Alice can obtain the true signature s of m by computing

$$s = s' \cdot r^{-1} \bmod n$$

Now Alice’s message m has a signature she could not have obtained on her own. Note, that even though the signature itself is secure provided that factoring remains difficult, the signature is still unconditionally “blind” since r is random.

Blind signature mechanisms are a fundamental element in untraceable digital cash systems and in electronic voting schemes. However, they have an implicit major drawback. Note that nothing prevents Bob from signing any message chosen by Alice. This means Bob could actually be required to sign very dangerous strings like “I, Bob, owe a million dollars to Alice”. To prevent such attacks, cut-and-choose techniques are used. Cut-and-choose techniques are based on sending to Bob a certain number, say p , of blinded messages. Bob chooses at random one of the received messages and requests Alice to reveal the blinding factor of all other messages. By checking that all $p - 1$ unblinded messages are inoffensive, Bob is convinced that the message that remains still blinded is inoffensive too. The probability of Alice successfully getting the signature of Bob on a malicious message is $1/p$, which can be made as small as desired.

4.3 Anonymous communications using mix-nets

A mix-net is a useful tool whenever unlinkability of sender and recipient is desired in a certain communication application. Mix-nets are currently mainly used to allow anonymous e-mail communications. Note that even if neither authentication nor digital signatures are used in a certain communication, traffic analysis is still possible. Communication protocols send data

encapsulated in packets, which contain concrete headers including source and destination addresses.

The concept of a mix was introduced by Chaum [Cha81]. Briefly, a mix is an entity that, in addition to forwarding incoming messages, hides the relationship between incoming and outgoing messages. This is done by grouping together a number of incoming messages, and shuffling them before forwarding them. In order to avoid content correlation between incoming and outgoing messages, all messages to a mix need to be encrypted in such a way that successive encryptions of the same message give different results. This can be easily achieved using enveloping procedures where a random session key is involved. It is also necessary to have all messages of uniform length. Otherwise, every outgoing message could be correlated to the incoming message(s) of the same length. Random padding is normally used for this purpose.

Using a mix as a common message forwarder for a group of communicating users, unlinkability of sender and recipient is assured to anyone except the mix itself. This means that the mix has to be honest. To relax this assumption, a mix-net is constructed, by cascading several mixes. Messages sent to a mix-net are successively enveloped to be readable only to the intended mix, in turn. Each mix receives messages from the previous hop (either a user—in the initial step— or another mix—in intermediate steps—). Digital envelopes are opened and the resulting contents are shuffled before being forwarded to the next mix, which operates in the same way. The last mix of the cascade performs the last shuffling and delivers messages to the final intended recipients. Using a mix-net, only a single mix of the cascade is required to be honest in order to prevent other mixes from breaking unlinkability of senders and recipients.

4.4 Vote-tags

The use of vote-tags was first devised in [Sak94], to allow open objections to the tally (i.e. to allow a voter, whose ballot has not been counted, to claim with no need to reveal in which way he actually voted). Nonetheless, vote-tags solve also a related problem, which is posed by the use of previously presented cut-and-choose techniques.

Briefly, a vote-tag is a piece of data which reveals no information of the vote itself but that is intrinsically linked to it. There is no formal definition of a vote-tag. Nevertheless, a vote-tag must fulfill two compulsory conditions:

1. Given a certain vote-tag alone, it has to be hard (or even better, impossible) to discover the corresponding vote.
2. A vote has to be bound to the corresponding vote-tag in such a way that it is hard to create a different valid vote without modifying the vote-tag. This statement assumes knowledge only of the vote and the vote-tag, and not of any other related information.

In [Sak94], vote-tags were presented as the public component of asymmetric key pairs randomly generated by voters during the vote casting protocol. If the vote itself is signed with the private key corresponding to the public key used as vote-tag, then the above conditions are satisfied. First, it is obvious that a random public key alone does not reveal any information about the vote. Second, to create another valid vote while keeping the corresponding original public key intact, requires knowledge of the private key used in the signature.

However, the use of public keys as vote-tags has a practical problem. The generation of a pair of asymmetric keys requires some time. It would not be a problem if only one vote-tag was needed to cast a ballot, but cut-and-choose techniques force the voter to generate many (ten, twenty, may be one hundred) pairs of asymmetric keys. This is not very convenient. For this reason, we suggest the use of one-way hash functions as vote-tags (see [Pre93] for a very comprehensive treatment of one-way hash functions). In this way, a vote-tag is just the result of applying a one-way hash function to the vote. To compute this value is many orders of magnitude faster than to generate an asymmetric key pair. Still, a vote-tag calculated as the result of a one-way hash function over the vote, fulfills the required conditions. This comes straightly from the properties of one-way hash functions.

5 Generic example

The following notation will be used in the presentation of the example:

- V : A particular voter.
- VC : Voting centre (ballot collecting authority).
- $S_{entity}(M)$: The digital signature of message M created with the private key of $entity$ over a digest of M . For our purposes, $S_{entity}(M)$ denotes both M in clear and the related signature bit-string.

- $M_1 \mid M_2$: Concatenation of messages M_1 and M_2 .
- *vote*: A data string which uniquely identifies one of the voting options.
- *vote-tag*: The vote-tag which *vote* is bound to.
- *ident*: A data string identifying the current election (e.g. the date).

Figure 1 summarizes the main steps involved in the voting phase of a typical electronic voting scheme. Normally, any interaction between voters and the voting centre starts with a bilateral authentication and the establishment of a security context for the voting session. In this way, step one of our generic voting scheme could consist of using the GSS-API protocol [Lin93] to achieve these objectives. Following the GSS-API protocol, voter and voting centre have authenticated each other and are able to exchange further data using confidentiality, authenticity and integrity services. Due to the initial authentication, the voting centre knows exactly who is contacting it. At this point, the electoral roll has to be consulted to check if that particular user is an eligible voter or not. In the negative case, the protocol is stopped. Otherwise, the user is allowed to vote. The voting procedure will consist, in summary, of obtaining a valid ballot for the current election (action which must require the involvement of the voting centre) and, later on, sending the ballot anonymously to the voting centre.

A possible way to obtain a valid ballot for the election would be to have the desired vote signed by the voting centre. By using a blind signature mechanism, this could be achieved while at the same time preventing the voting centre from knowing the vote. However, cut-and-choose techniques would break this privacy. To overcome this inconvenience, the voting centre has to be required to blindly sign a vote-tag, instead of the vote itself. In this way, after the voter has chosen the vote to be cast, he constructs an adequate vote-tag (for example, he computes a hash value of the vote). Afterwards, an identifier of the current election is appended to this vote-tag. This whole data structure is blinded, as explained in previous section, and sent to the voting centre in step two. Actually, the voter generates (and blinds and sends) as many adequate vote-tags as required by the cut-and-choose technique. This is easily reached by, for example, appending different random paddings to the vote before computing the hash value of it. Note that the fact of unblinding all but one vote-tags does not allow the voting centre to know in which way the voter is voting. Finally, one of the vote-tags is blindly signed by the

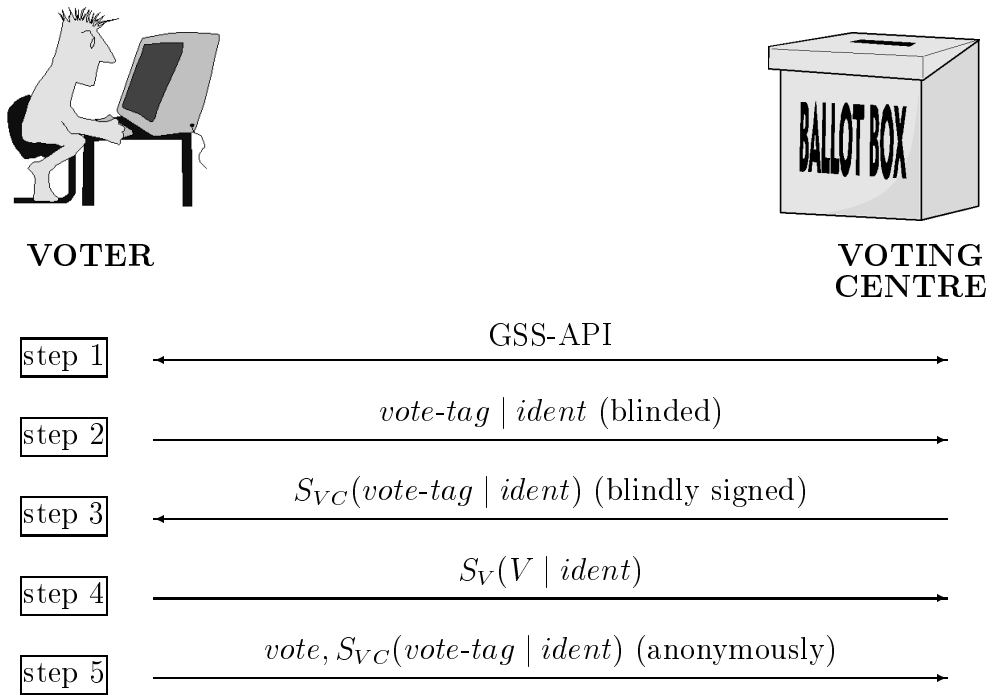


Figure 1: Generic example of ballot casting procedure.

voting centre and sent back to the voter, in step three. After this is done, the voting centre should update the electoral roll, crossing off that particular voter's entry. If the same voter tries to contact again the voting centre, his connection will be refused.

At this point of the protocol, the voter owns a proof that his vote is a valid vote for the current election. Such proof consists of the voting centre's signature over a hash function of the vote, together with the election's identifier. In the following, we will call both data strings, vote and signed vote-tag, as a validated ballot. Once the voter owns a validated ballot, he is required to send a proof of this fact to the voting centre. For this reason, in step four, the voter sends to the voting centre a signed string containing his identity and the election's identifier. The usefulness of this data will become clear in next section.

Validated ballots have to be sent to the voting centre. However, such communication step (number five on the Figure) must be made somehow anonymously. A very simple approach could be to use voting booths (i.e. several fixed computer terminals on the network) from where voters could cast their ballots in an absolutely anonymous way. However, this approach (in addition of requiring two sessions) clearly violates the mobility of voters. For these reasons, mix-nets are normally used to allow voters to send the ballots to the voting centre, from their own usual network addresses. By using a mix-net, the voting centre is not able to correlate received ballots with original ballots introduced to the mix-net by voters. Still, the voting centre is able to verify that the received ballots are valid for the current election because of its own signature over the accompanying vote-tags.

During the voting phase, the described protocol is executed by every voter and the voting centre. Following the end of this phase, the construction of the tally has to be initiated. Very frequently, the references on electronic voting schemes do not make concrete about *who* is really deciding when the voting centre stops accepting ballots and starts the counting process. More realistic is the scheme in [BR96], since it introduces the definition of an electoral board that operates the voting centre. The electoral board is constituted as a subset of voters who follow specific procedures to manage the voting centre during every voting phase. A secret sharing scheme is used to avoid dishonest members of the board affecting the election.

After the voting centre has tabulated adequately all ballots, the tally must be published on the network. Following our generic example, the published tally would comprise two separate lists, according to the format shown in

use of vote-tags in this way allows to do open (public) objections to the tally, which means that the voter is able to prove the fraud with no need to reveal the vote cast. As a conclusion, validated ballots cannot be altered, not even by the voting centre. Certain forms of vote-tags (e.g. public keys, as suggested in [Sak94]) allow the voter to alter *his own* ballot before casting it (i.e. before step five). This means, in practice, that the voter could change his mind within a given period of time. Other forms (e.g. hash functions as suggested in this report) do not allow even the voter to alter his own ballot.

If a validated ballot is eliminated from the final tally, the affected voter is able to do an open objection, following the same procedure as before.

Finally, note that only the voting centre could be able to add invalid ballots to the tally because it is the only entity that is authorized to sign vote-tags. The voting centre could therefore create apparently valid ballots and in this way forge the tally. A possible way to prevent the voting centre from adding ballots on its own, consists of publishing the list of ballots together with the list of voters who have voted (data collected from voters during step four of the presented protocol). In this way, the number of entries in the list of ballots should be equal than the number of voters who have contacted the voting centre. If there are more ballots than voters, it means cheating by the voting centre (assuming that all voters really accomplished step four of the protocol). However, if the number of tabulated ballots is smaller than the number of voters who obtained a valid ballot, it could either mean that some voters did not bother voting finally, or that the voting centre has removed validated ballots from the tally. In the presented example of voting scheme, there is no way to find out who is really the cheater. In such conditions, some assumptions have to be made (for example, it has to be assumed that any voter who obtains a validated ballot, will finally cast it). This kind of assumptions (even though not desired) are not uncommon in electronic voting schemes because of the difficulties of achieving all security requirements simultaneously. Actually, one of the goals of researchers in this area is to restrict the number of such assumptions.

6.2 Democracy

Democracy has to be assured through proper authentication of voters to the voting centre, which handles the electoral roll. Therefore, access control to the file that implements the electoral roll is needed, too. The format and structure of this file varies from voting scheme to voting scheme. It may

be implemented as a locally stored database with an application-dependent format. On the other side, it may be supported by a distributed, open and standardized information service, such as the ITU-T X.500 Directory Service [CCI88a]. Be that as it may, secure access to the electoral roll has to be guaranteed. In this way, authorization policies may be enforced. Only the adequate voting authority must be able to add eligible voters to the electoral roll, and only the ballot collecting centre must be able to set the boolean attribute which indicates that a particular voter has already voted.

Furthermore, in the proposed example, the voting centre itself can also be prevented from attacking the democracy property without being detected. However, in order to reach this feature, the assumption that any voter who obtained a valid ballot will finally cast it, must be made. Given such assumption, if the voting centre signs vote-tags for non-eligible users (or signs more than one vote-tag to the same eligible voter), this will be reflected when the tally is published. The list of ballots would contain more entries than the list of eligible voters who voted. Therefore, these cases can be treated analogously as if the voting centre added ballots on its own.

6.3 Privacy

Privacy consists of three properties: anonymity, uncoercibility and fairness. We will discuss each in turn.

6.3.1 Anonymity

Anonymity is probably the cornerstone of electronic voting schemes. We have defined anonymity of ballots as the property that nobody is able to link any ballot to the voter who has cast it. In practice, this requirement is however normally relaxed and there is usually someone who is able to link ballots to voters. Actually, there are no obvious solutions to anonymity since it is, in some way, contradictory to the democracy principle. How can the voting centre receive ballots from very well identified voters and still be not able to link these ballots to the corresponding voters?

Electronic voting schemes can be usually classified into two main groups, according to how they answer previous question. First group, less numerous, is well represented by its earliest proposal [BY86]. The fundamental principle of these voting schemes is to break the voting centre into different parts. Anonymity is assured to the extent that all parts will not conspire against the

voter. Individual ballots are divided up among the different parts of the voting centre. Each part tallies the piece of ballot it receives (there are protocols to verify that the tally is correct), and the final result of the election comes from the sum of all tallies. However, this group of electronic voting schemes shows several practical problems. Basically, they require great amount of communication and processing to cast a single ballot. Moreover, votes are restricted to a yes/no format. For these reasons, most of the proposed electronic voting schemes belong to the second group.

Second group of voting schemes relies on the existence of an anonymous channel between voters and the voting centre. Normally, such anonymous channel takes the form of some kind of mix-net as defined in Section 4. Use of an anonymous channel to cast ballots requires that every ballot includes an authorization from the voting centre, to preserve democracy. Again, there is no single solution to the way of constructing these authorizations. There are mainly four alternatives. First alternative consists of breaking the voting centre into two different agencies, one issuing authorizations and the other receiving ballots. However, if these two agencies collude, anonymity is defeated. The second alternative uses physically protected voting booths and zero-knowledge proofs of knowledge [GMR85]. However, use of voting booths in electronic voting schemes is not desired at all, as it will be explained later. Third alternative, well represented by [NSS91], is based on the all-or-nothing disclosure of secrets (ANDOS) protocol [BCR86]. This protocol is used in order to anonymously distribute authorizations among voters. However, the complexity of the ANDOS protocol is much greater than that of the technique used by the fourth alternative, while the benefits are exactly the same. The fourth manner of issuing valid authorizations is through the use of blind signature mechanisms (technique explained in Section 4). Due to the simplicity of this technique, the most widely accepted way of achieving both democracy and anonymity in electronic voting schemes is by combining blind signatures and, at some stage of the voting process, an independent mix-net. This is the alternative chosen for our example.

Note that if a mix-net is used as anonymous channel between voters and voting centre, at least one of the mixes must remain honest. If all mixes collude, they are able to defeat anonymity. For this reason, the announced anonymity requirement becomes in practice slightly relaxed. Since at least one of the components of the mix-net must be trusted, some voting schemes even adopt the approach of skipping the use of any anonymous channel and directly trusting the voting centre. The voting centre would publish all bal-

lots after the voting period, without making public the origin of each ballot. Evaluation and certification of the hardware and software components of the voting centre become compulsory in those cases, together with a certain degree of tamper-proofing.

6.3.2 Uncoercibility

The second privacy factor was introduced in [BT94] and named uncoercibility. Voters can be coerced basically in two ways. Either votes may be bought or voters may be intimidated in order to cast a particular vote. Both processes require the voter to prove the coercer in which way he voted. This is straightforward in almost all electronic voting schemes due to the *voting receipts* given at voting time by the voting centre to voters (step three of our example). As we have seen, a voter uses his receipt to prove forgeries on the tally if his ballot has not been properly counted (preservation of the accuracy requirement). However, the same receipt can also be used by a coercer or vote buyer to obtain proof of the vote cast. Accuracy somehow contradicts uncoercibility.

If voters are not able to prove in which way they voted, there is no possibility for a third party to coerce them with certainty of success. Such approach has led to receipt-free systems. Receipt-freeness of electronic voting schemes has been studied in [BT94], [NR94], and [SK95]. However, all three proposals are based on cumbersome physical assumptions, not far from those encountered in traditional elections. Voters are required to cast their ballots from a physically isolated voting booth or to use untappable communication channels. Note that the latter also forces the voter to be located at some particular place. Such physical assumptions do not fit well with the notion of practical electronic voting schemes to be utilized over computer networks. In particular, the mobility of voters is clearly not considered. In fact, the only advantage gained over traditional methods is the increment of both speed and ease during the ballot counting process. Nonetheless, this can also be reached just by changing the traditional paper-based ballots by magnetic card ballots read by an electronic ballot box.

More recently, in [RBR98] we introduced the concept of *receipt-hidden schemes*, to overcome the inconveniences of receipt-free schemes. In this way, we presented an uncoercible voting scheme which does not restrict at all the mobility of voters. This is achieved through the use of tamper-resistant smartcards which, in some way, act as mobile physically protected voting

booths. Anyway, uncoercibility seems to need some special hardware to be solved. For clarity, we have not included in the proposed example any of the techniques used to provide uncoercibility. As a consequence, the example becomes a coercible voting scheme. Note that the voter has a proof of the vote cast. If vote-tags are implemented as a public key, the proof consists of the related private key, and lasts for ever. If vote-tags are implemented as one-way hash functions, a voter can prove to a coercer in which way he voted only by successfully guessing the exact bit-string that constitutes his ballot, prior to the publication of ballots. Therefore, the proof lasts in this case only until the publication of the tally, but not after.

6.3.3 Fairness

The property of fairness, introduced by [FOO92], requires that all ballots stay secret until the voting phase has ended. The objective is to prevent anyone (normally the voting centre) from knowing intermediate results of the voting. Such knowledge could be used to affect other voters' behavior. The voting scheme in [FOO92] solves fairness by using a bit-commitment protocol [Nao89]. Such protocol assures that a voter cannot change his mind after his ballot is cast, while it prevents the voting centre from having knowledge of the ballot as long as the voter desires. However, such scheme has some drawbacks. First, the voter is involved in two anonymous sessions with the voting centre. Second, and worse, a voter can cheat the bit-commitment protocol and there is no way to decide whether the cheater is the voter or the voting centre.

In our example, fairness could be (not very conveniently) assured by requiring all mixes of the mix-net to delay forwarding ballots until the voting phase is ended. Again, a single mix honest assures that the voting centre does not receive ballots until the end of the voting phase. Fairness would be defeated if all mixes of the mix-net, together with the voting centre, collude.

6.4 Verifiability

Verifiability of electronic voting schemes is usually very related to accuracy. In the presented example, verifiability is assured because ballots that are published in the final tally contain both the vote and the signed vote-tag. In this way, every voter just has to look in the tally for his own vote-tag. In case his ballot has not been properly counted, the voter is able to do an open

objection, as it has already been commented. However, note that this kind of objections could in practice reveal in which way the voter voted. Even though the exact vote itself is not revealed, it can be more or less deduced from the conjecture that one would not trouble to make objection if the result of the election is favorable, despite of the treatment of his vote. To solve this problem, [Sak94] proposes to separate the time to disclose accepted ballots and the time to announce the result of the election. The proposed idea is based on announcing the accepted ballots encrypted by the voting centre's public key prior to publishing the tally itself. In this way, even if a voter makes an objection, it purely means his ballot is not counted, and not because he does not like the result.

7 Implementation

So far, electronic voting schemes have been presented mainly from a theoretical point of view, without considering too many practical issues. Actually, some of the current proposals of electronic voting schemes are still designed without caring about the complexity of their possible real implementations. Consequently, the computation and communication costs usually prevent the development of a practical product. Currently, there are not too many already implemented electronic voting schemes. Among them, [BR96], [CC96], and [Hwa96] are counted. However, even though the implementation of electronic voting schemes is still not mature, [CC96] exposes three extra properties desired in the case of implementable schemes:

- **Convenience:** A voting scheme is convenient if it allows voters to cast their ballots quickly, in one session, and with minimal equipment or special skills.
- **Flexibility:** A voting scheme is flexible if it allows a variety of ballot question formats including open ended questions.
- **Mobility:** A voting scheme is mobile if there are no restrictions on the location from which a voter can cast a ballot.

Traditionally, electronic voting schemes assume restricted environments involving a single voting centre. Therefore, the number of voters is not expected to be large and the implementations (where they exist) are typically

operated on a LAN. A single voting centre is clearly not enough when the number of voters is potentially very large (nation-wide elections, for example). The ballot box and the electoral roll files could grow to unmanageable proportions. Even worse, the voting centre itself would become a bottleneck. Therefore, in addition to the mentioned properties, we suggest a new extra property needed in case of large scale elections:

- **Scalability:** A voting scheme is scalable if it allows any number of voters to participate in the election.

In [RBR97] we offered a solution to the scalability of electronic voting schemes based on the concurrent operation of a set of *electoral colleges*. Each electoral college acts as a voting centre, receiving ballots only from a small group of registered voters. However, such approach requires some mechanisms to assure the coordination of all electoral colleges. Several open questions appear that must be addressed. To list a few examples, distribution of voters while assuring the democracy property, coordination of the opening and closing times assuring fairness, management of a distributed electoral roll, or secure computation of the global tally from partial results.

The hierarchical topology shows to be an effective way to help solve these problems, while keeping the system fully scalable. In Figure 3, an example of voting hierarchy is depicted. The leaves of the voting tree represent electoral colleges. Intermediate nodes act as *counting centres*. The main task of a counting centre is to join all partial tallies corresponding to its subtree. Outcomes are delivered bottom-up through the tree until they reach the root, which performs the computation of the global tally and publishes it. This last entity is not only a counting centre, but it also creates and maintains the electoral roll, and it does some administrative and security tasks related to the election. Actually, it represents the electronic version of the official organization that is in charge of conventional elections in most countries. We named it the *national voting authority*.

ITU-T X.500 Directory Service [CCI88a] plays a central role in the proposed scalable electronic voting scheme. It allows the distribution of a large electoral roll in a reliable, standardized, and secure manner. Furthermore, the whole coordination of voting authorities is strongly based on secure access to a collection of specifically defined object entries stored in the Directory. The hierarchical model, supported by X.500 Directory, allows to design an implementable solution which may be operated over a WAN or a group of

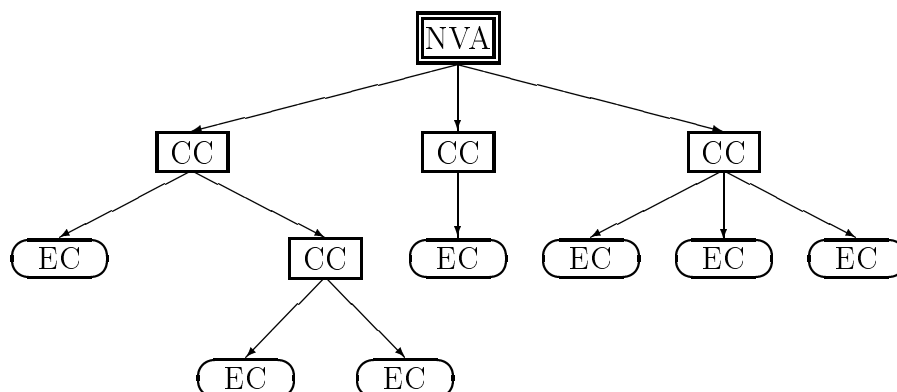


Figure 3: Hierarchical arrangement of voting authorities.

interconnected networks. Actually, we are currently developing a prototype to be operated as a large scale electronic voting scheme over the Internet. A first release is due for year 2000.

References

- [BCR86] Brassard, G. and Crépeau, C. and Robert, J.M. (1986) All-or-Nothing Disclosure of Secrets, in *Advances in Cryptology – CRYPTO '86*, LNCS **263** (Springer-Verlag), 234–238.
- [BR96] Borrell, J. and Rifà, J. (1996) An Implementable Secure Voting Scheme. *Computers & Security*, **15**, 327–338.
- [BT94] Benaloh, J. and Tuinstra, D. (1994) Receipt-Free Secret-Ballot Elections, in *26th annual ACM Symp. on Theory of Computing*, (ACM), 544–553.
- [BY86] Benaloh, J.C. and Yung, M. (1986) Distributing the Power of a Government to Enhance the Privacy of Voters, in *5th annual ACM Symp. on Principles of Distributed Computing*, (ACM), 52–62.
- [CC96] Cranor, L.F. and Cytron, R.K. (1996) Design and Implementation of a Practical Security-Conscious Electronic Polling System. *Washington University Report WUCS-96-02*.

- [CCI88a] CCITT (1988) Recommendation X.500: The Directory – Overview of Concepts, Models and Services.
- [CCI88b] CCITT (1988) Recommendation X.509: The Directory – Authentication Framework.
- [Cha81] Chaum, D. (1981) Untraceable Electronic Mail, Return Addresses and Digital Pseudonyms. *Communications of the ACM*, **24**, 84–88.
- [Cha83] Chaum, D. (1983) Blind Signatures for Untraceable Payments, in *Advances in Cryptology – CRYPTO '82*, (Plenum Press), 199–203.
- [Cha85] Chaum, D. (1985) Security without Identification: Transaction Systems to Make Big Brother Obsolete. *Communications of the ACM*, **28**, 1030–1044.
- [DOW92] Diffie, W. and Van Oorschot, P.C. and Wiener, M.J. (1992) Authentication and Authenticated Key Exchanges. *Designs, Codes and Cryptography*, **2**, 107–125.
- [FOO92] Fujioka, A. and Okamoto, T. and Ohta, K. (1992) A Practical Secret Voting Scheme for Large Scale Elections, in *AUSCRYPT '92*, LNCS **718** (Springer-Verlag), 244–251.
- [GMR85] Goldwasser, S. and Micali, S. and Rackoff, C. (1985) The Knowledge Complexity of Interactive Proof Systems, in *17th annual ACM Symp. on Theory of Computing*, (ACM), 291–304.
- [Hwa96] Hwang, J.J. (1996) A Conventional Approach to Secret Balloting in Computer Networks. *Computers & Security*, **15**, 249–263.
- [Lin93] Linn, J. (1993) Internet RFC **1508**: Generic Security Service Application Program Interface.
- [Nao89] Naor, M. (1989) Bit Commitment Using Pseudo-Randomness, in *Advances in Cryptology – CRYPTO '89*, LNCS **435** (Springer-Verlag), 128–136.
- [NR94] Niemi, V. and Renvall, A. (1994) How to Prevent Buying of Votes in Computer Elections, in *ASIACRYPT '94*, LNCS **917** (Springer-Verlag), 141–148.

- [NSS91] Nurmi, H. and Salomaa, A. and Santean, L. (1991) Secret Ballot Elections in Computer Networks. *Computers & Security*, **10**, 553–560.
- [Pre93] Preneel, B. (1993) Analysis and Design of Cryptographic Hash Functions. *Ph.D. dissertation*, Katholieke Universiteit Leuven.
- [RBR97] Riera, A. and Borrell, J. and Rifà, J. (1997) Large Scale Elections by Coordinating Electoral Colleges, in *IFIP SEC '97, Information Security in Research and Business* (Chapman&Hall), 349–362.
- [RBR98] Riera, A. and Borrell, J. and Rifà, J. (1998) An Uncoercible Verifiable Electronic Voting Protocol, to appear in *IFIP SEC '98* (Chapman&Hall).
- [RSA78] Rivest, R.L. and Shamir, A. and Adleman, L.M. (1978) A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of the ACM*, **21**, 120–126.
- [Sak94] Sako, K. (1994) Electronic Voting Scheme Allowing Open Objection to the Tally. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences Vol.E77-A*, **1**, 24–30.
- [SK95] Sako, K. and Kilian, J. (1995) Receipt-Free Mix-Type Voting Scheme, in *EUROCRYPT '95*, LNCS **921** (Springer-Verlag), 393–403.