List Signature Schemes and Application to Electronic Voting

Sébastien Canard and Jacques Traoré

France Telecom R&D
42 rue des Coutures - BP 6243
F-14066 Caen Cedex 4, France
{sebastien.canard,jacques.traore}@francetelecom.com

Abstract

An electronic voting scheme is a mechanism in which voters can securely vote and several authorities can collect these votes to calculate the result. Several voting schemes exist ([3, 4, 14]) and most of them are based on three cryptographic concepts: mix-nets, blind signatures or homomorphic encryption. Another potential primitive for electronic voting is given by group signature schemes, which allow a group member to anonymously sign messages on behalf of the group. Unfortunately, such signature schemes are not directly usable in electronic voting. This is why we introduce in this paper a variant of group signature schemes, called (electoral) list signature schemes, in which the signatures cannot be opened by anybody and are linkable if (and only if) they have been produced within the same "sequence" (typically an election day). We give a concrete example of a list signature scheme. Then, we design an electronic voting scheme based on this new concept. Our solution is surprisingly simple (only one connection to a database) and efficient (only a few proofs of knowledge), while satisfying the fundamental needs of security in electronic voting. Finally, it is a very convenient solution for both on-line voting (via Internet for example) and off-line voting (using electronic voting booths).

1 Introduction

Group signature schemes have been introduced in 1991 by Chaum and van Heyst [10]. They allow members to sign a document on behalf of the group in such a way that the signatures remain anonymous and untraceable for everyone but a designated authority, who can recover the identity of the signer whenever needed (this procedure is called "signature opening"). Moreover, this type of signature must be unlinkable for everybody but the authority (no one else can decide whether two different valid signatures were computed by the same group member or not). There are many applications to group signature schemes and folklore even claims that they can be applied to electronic voting ([1]), each voter signing his vote with a group signature. However, this claim seems wrong as two group signature properties conflict with electronic voting requirements. First of all, a vote should never be "open" since nobody should know the vote of any voter. Second, if two signatures by the same voter were unlinkable, then he could produce several votes for the same election.
This is the reason why we modify security properties of group signature schemes so as to obtain a variant called (electoral) list signature schemes. This sort of signature achieves the same properties as group signature schemes except two. First, the signatures cannot be

"opened" by anybody. Second, it is possible to link two signatures of the same person during a so-called sequence (for example an election day), while unlinkability related to distinct sequences remains preserved.
So the first contribution of this paper is to introduce the notion of list signature scheme, by describing the different phases and the security properties. Then we describe a scheme that fits all these properties, based on Ateniese et al. group signature scheme [1].

An electronic voting scheme is a protocol allowing voters to securely vote by interacting with a set of authorities who collect the votes and calculate the result of the election. Also, there are some basic security requirements for such schemes: protection of the confidentiality of voters (anonymity), detection of voters' cheating (two votes by the same voter, duplication of votes) and detection of authorities' cheating (tampered results, publication of partial results). Moreover, there are two types of electronic voting: on-line voting (or remote voting, for example via Internet) and off-line voting (by using a voting machine or an electronic polling booth).
Several papers present schemes for electronic voting ([3, 4, 14]). Most of them use (at least) one of the following cryptographic primitives: mix-nets ([8]), blind signatures ([9]) or homomorphic cryptosystems. There are also other papers such as [15] which uses a variant of group signature schemes similar to our list signatures. However authors of [15] seem to use unpractical group signature schemes (Chaum and van Heyst's ones). Furthermore, their scheme requires each voter to connect several times to various authorities, and the possibility to make a group signature is valid for one election only. Finally, their "linkable" group signatures can be "opened".
In this paper, we suggest a new electronic voting scheme which is surprisingly simple once given a list signature scheme. Our solution requires only one voter's connection for each election (versus three connections for the blind signature solution) and is very efficient (only a few proofs of knowledge, contrary to the homomorphic cryptosystem solution) while meeting all the security requirements introduced above.

The paper is organized as follows. Section 2 defines the new concept of list signature schemes, presents an example of such a scheme and studies its security. Section 3 describes the new electronic voting scheme and studies its security. Finally, we conclude in Section 4.

2 List Signature Schemes

This section presents a new type of signature scheme, which is different from group signature schemes in that (1) they cannot be "opened" and (2) they are linkable all along a sequence. A very close variant of our list signature schemes has been proposed in [6] but without the notion of sequence (their proposal corresponds to group signatures that are linkable and optionally non-"openable").

2.1 Definitions and Properties

A list signature scheme implies various entities: a Trusted Authority (TA) who will register new list members, a Revocation Authority (RA) who will be responsible for the revocation of list members and users (U_i) who will be list members.
The concept of list signature scheme can be defined as follows:

Definition 1. *A list signature scheme is a digital signature scheme that consists of the following algorithms:*
- Setup: generation of parameters and public keys.
- Join: protocol between a new user and TA that permits the new user to obtain his signature capability (made of a private key and a certificate).
- Revocation: RA removes the signature capability of a revoked user.
- Update: each valid user modifies his own keys after the modification of the set of users (i.e. after an execution of the Join or Revocation algorithm).
- Organization of a Sequence: generation of specific parameters to a particular sequence (and only this one).
- Signature: a valid user can anonymously sign a message of his choice within a particular sequence.
- Verification: everybody can check the validity of a signature made by a valid user within a particular sequence.

Definition 2. *A secure list signature scheme satisfies the following properties:*
(i) Correctness: a signature produced by a valid user is always valid.
(ii) Unforgeability: only valid users are able to sign messages.
(iii) Anonymity: given a valid list signature, it is infeasible for anyone to identify the actual signer.
(iv) Partial Linkability: within a sequence, anybody is able to say if two valid signatures were produced by the same user.
(v) Partial Unlinkability: deciding whether two valid signatures from two different sequences were computed by the same valid user is infeasible.
(vi) Coalition-Resistance: a colluding subset of group members should not be able to generate a valid signature.
(vii) Revocability: RA is always able to remove signature capability of any user.

2.2 An Example of List Signature Scheme

Our proposed list signature scheme is based on the Strong-RSA assumption (independently introduced by Barić and Pfitzmann [2] and by Fujisaki and Okamoto [13]), and on the decisional Diffie-Hellman assumption in groups of unknown order. It is based on the proposal of Ateniese et al. [1] using the revocation mechanism of Camenisch and Lysyanskaya [7].
Throughout the paper, the symbol $\|$ will denote the concatenation of two strings. The symbol $\emptyset$ will denote the empty string. The notation "$x \in_R E$" means that x is chosen uniformly at random from the set E. Moreover, $PK(\alpha / f(\alpha, \ldots))(m)$ will be a signature of knowledge on message m of a value α that verifies the predicate f (we call a signature of knowledge a signature derived from a zero-knowledge (honest-verifier) proof of knowledge using the Fiat-Shamir heuristic [12]). Finally, the set $QR(n)$ will correspond to the set of all quadratic residues modulo n.

2.2.1 Setup Protocol

We have to introduce some security parameters. We will need integers $\epsilon > 1$, k and l_p. Moreover, λ_1, λ_2, γ_1 and γ_2 will denote lengths (see [1] for more details). Let us define $\Lambda =]2^{\lambda_1} - 2^{\lambda_2}, 2^{\lambda_1} + 2^{\lambda_2}[$ and , $=]2^{\gamma_1} - 2^{\gamma_2}, 2^{\gamma_1} + 2^{\gamma_2}[$. Furthermore, we will need a collision-resistant hash function $H : \{0,1\}^* \rightarrow \{0,1\}^k$.

The initial phase involves TA and RA generating the set of keys:

TA:	- Select distinct primes p'_1 and q'_1 (size l_p), such that $p_1 = 2p'_1 + 1$ and $q_1 = 2q'_1 + 1$ are primes. - Compute the modulus $n_1 = p_1 q_1$. - Choose random elements a, a_0, b, g, $h \in_R QR(n_1)$ (of order $p'_1 q'_1$).
RA:	- Select distinct primes p'_2 and q'_2 (size l_p), such that $p_2 = 2p'_2 + 1$ and $q_2 = 2q'_2 + 1$ are primes and such that n_1 and $n_2 = p_2 q_2$ are relatively prime. - Choose a random element $u \in_R QR(n_2)$ (of order $p'_2 q'_2$). - Set up (empty for now) E_{add} and E_{del}.

At the end of this protocol, we note the list public key $PK = (n_1, n_2, a, a_0, b, g, h, u)$, the TA's private key $SK_{TA} = (p'_1, q'_1)$ and the RA's private key $SK_{RA} = (p'_2, q'_2)$.

2.2.2 Join Protocol

This algorithm is based on the Ateniese et al. one ([1]) using the revocation mechanism of [7] and all zero-knowledge proofs of knowledge can be found in this paper. It takes into account that there are two distinct authorities (and therefore two distinct moduli), one for the list certificate and one for the revocation.
During this protocol, a new list member U_i will obtain from TA and RA a private key x_i and a member certificate $[A_i, e_i, u_i]$ such that $a^{x_i} a_0 = A_i^{e_i} \pmod{n_1}$ and $u_i^{e_i} = u \pmod{n_2}$.
At the end of the Join protocol, TA creates a new entry $\{[A_i, e_i, u_i]$,Identity of $U_i\}$ in the membership database and RA keeps in mind the certificate for further modification of the user list (see Sect. 2.2.3) and modifies u in the following way: $u := u^{e_i} \pmod{n}$. The whole join protocol is similar to the combination of the one of [1] and of [7].

2.2.3 Revocation/Update Protocols

In case of a revocation of the list member U_k (whose certificate is $A_k, e_k, u_k)$), RA has to modify u in the following way: $u := u^{1/e_k} \pmod{n_2}$. He then modifies $E_{del} := E_{del} \cup \{e_k\}$ and publishes each value.
After a Join protocol, all members U_j (using E_{add}) have to make the following modification of their key: $u_j := u_j^{e_i} \pmod{n_2}$.
After a Revocation protocol, each user $U_j \neq U_k$ (using E_{del}) has to do the following computations: $u_j := u_j^b u^a \pmod{n_2}$ where a and b are such that $ae_j + be_k = 1$.
In both cases, the user U_j always knows a couple (e_j, u_j) such that $u_j^{e_j} = u \pmod{n_2}$.

2.2.4 Organization of a Sequence

Before the beginning of a new sequence, TA publishes the representative of the sequence, that is a random integer $m \in_R QR(n_1)$. This is a public integer that will be used by every signer during this sequence. The representatives of different sequences must be uncorrelated. In particular, it must be infeasible to compute the discrete logarithm of one representative w.r.t. the one of any other sequence.
A simple solution can be to compute $m = (H'(date))^2 \pmod{n_1}$ where $date$ is, for example, the date of the beginning of the sequence and $H' : \{0,1\}^* \rightarrow \{0,1\}^{2l_p}$ is a collision-resistant

hash function. Consequently, everybody can verify the validity of this random element by performing the same computations.

2.2.5 Signature

During the signature phase, the user U_i proves that he knows a membership certificate and that he uses the representative of the sequence. The main tool is a proof of knowledge of a discrete logarithm with respect to different modulus (see e.g. [5]).

First of all, U_i generates random values $w, w_1, w_2, w_3 \in_R \{0,1\}^{2^l p}$ and then computes:
(i) $T_1 = A_i b^w \pmod{n_1}$ (ii) $T_2 = g^w \pmod{n_1}$
(iii) $T_3 = g^{e_i} h^w \pmod{n_1}$ (iv) $T_4 = m^{x_i} \pmod{n_1}$
(v) $T_5 = g^{e_i} h^{w_1} \pmod{n_2}$ (vi) $T_6 = u_i h^{w_2} \pmod{n_2}$
(vii) $T_7 = g^{w_2} h^{w_3} \pmod{n_2}$
We can modify them[1], and then U_i has to make the following signature of knowledge:
$U = PK(\alpha, \beta, \gamma, \delta, \zeta, \eta, \theta, \iota, \kappa / a_0 = T_1^\alpha / (a^\beta b^\gamma) \pmod{n_1} \wedge 1 = T_2^\alpha / g^\gamma \pmod{n_1} \wedge T_2 = g^\delta \pmod{n_1} \wedge T_3 = g^\alpha h^\delta \pmod{n_1} \wedge T_4 = m^\alpha \pmod{n_1} \wedge T_5 = g^\alpha h^\zeta \pmod{n_2} \wedge T_7 = g^\eta h^\theta \pmod{n_2} \wedge u = T_6^\alpha / h^\iota \pmod{n_2} \wedge 1 = T_7^\alpha / (g^\iota h^\kappa) \pmod{n_2})(M)$.
The list signature of M is then $(T_1, T_2, \ldots, T_7, U)$.

2.2.6 Verification

A verifier V can check the validity of a signature $(T_1, T_2, \ldots, T_7, U)$ by simply verifying the signature of knowledge.
Moreover, everyone who has access to all signatures for a particular sequence can easily see if a user has signed twice or more by observing the value of T_4. The user cannot cheat (by using another value) because T_4 is linked with T_1 by the proof of knowledge and the private key x_i (remember that the equation implying T_1 can also be written $T_1^{e_i} = a_0 a^{x_i} b^{w e_i} \pmod{n_1}$).

2.2.7 Security of the Proposed List Signature Scheme

In this section, we examine the security of our list signature scheme. We first need to introduce the two following theorems.

Theorem 1. ***(Coalition-resistance)***
Under the Strong-RSA Assumption, a certificate of the list signature scheme $[A_i, e_i]$ *with* $x_i \in \Lambda$, $e_i \in$, *and* $A_i = (a_0 a^{x_i})^{1/e_i} \pmod{n_1}$ *can be generated only by TA provided that the number K of certificates that TA issues is polynomially bounded.*

As the Join algorithm is similar to the one describe by Ateniese et al., the proof of this theorem directly follows the one of their theorem (see [1] for more details). We now state the main theorem of this section:

[1] Using $A_i^{e_i} = a_0 a^{x_i} \pmod{n_1}$ and $u_i^{e_i} = u \pmod{n_2}$, (i) can also be written as $T_1^{e_i} = A_i^{e_i} b^{w e_i} = a_0 a^{x_i} b^{w e_i} \pmod{n_1}$ and then $a_0 = T_1^{e_i} / (a^{x_i} b^{w e_i}) \pmod{n_1}$; (ii) can also be written as $T_2^{e_i} = g^{w e_i} \pmod{n_1}$ and then $1 = T_2^{e_i} / g^{w e_i} \pmod{n_1}$; (vi) can also be written as $T_6^{e_i} = u_i^{e_i} h^{w_2 e_i} = u h^{w_2 e_i} \pmod{n_2}$ and then $u = T_6^{e_i} / (h^{w_2 e_i}) \pmod{n_2}$ and (vii) can also be written as $T_7^{e_i} = g^{w_2 e_i} h^{w_3 e_i} \pmod{n_2}$ and then $1 = T_7^{e_i} / (g^{w_2 e_i} h^{w_3 e_i}) \pmod{n_2}$.

Theorem 2. ***(Existential Unforgeability)***
Under the Strong-RSA Assumption and in the random oracle model, the signature protocol is existentially unforgeable against adaptative chosen messages attacks.

The list signature scheme is based on a zero-knowledge (honest verifier) proof of knowledge and we can conclude the theorem using the result of [16].
We can now study the security of our proposal by giving evidence that our scheme verifies all properties introduced in Definition 2.
(i) Correctness: by inspection.
(ii) Unforgeability: this is a consequence of Theorems 1 and 2 and of the fact that neither TA nor RA can create a new database entry if one of them does not want to (TA cannot compute u_i and RA cannot compute A_i).
(iii) Anonymity: as Theorem 2 holds and as all commitments are statistically hiding ones (see [7] for details), the only way to break the anonymity is to determine, from a certificate $[A_i, e_i, u_i]$, if the three discrete logarithms $\log_b T_1/A_i$, $\log_g T_2$ and $\log_h T_3/g^{e_i}$ and if the two discrete logarithm $\log_m T_4$ and $\log_a C_2$ are respectively equal. This is infeasible under the Decisional Diffie-Hellman assumption.
(iv) Partial Linkability: the signer cannot forge $T_4 = m^{x_i} \pmod n$ since he/she is obliged to use m and his own x_i which is linked to $T_1 = A_i b^w = (a_0 a^{x_i})^{1/e_i} b^w \pmod n$ by the signature (see Theorem 2).
(v) Partial Unlinkability: this is a valid property under the hypothesis that each representative of each sequence is chosen independently of former ones (in our case, finding the discrete logarithm of the representative of a sequence in the base of ancient ones must be infeasible). Then, if m and $\tilde{m}$ are two different representatives of two different sequences, a user U_i will compute $T_4 = m^{x_i} \pmod n$ and $\tilde{T}_4 = \tilde{m}^{x_i} \pmod n$. By the Decisional Diffie-Hellman Assumption, it is infeasible to learn anything about x_i. Furthermore, as the Theorem 2 holds and as the Sign protocol implies statistically hiding commitments (see [7]), the problem of linking two signatures can be reduced to the problem of deciding whether the three discrete logarithm $\log_b T_1/\tilde{T}_1$, $\log_g T_2/\tilde{T}_2$ and $\log_h T_3/\tilde{T}_3$ are equal. This is infeasible under the Decisional Diffie-Hellman assumption.
(vi) Coalition-Resistance: this follows from Theorems 1 and 2.
(vii) Revocability: the revocation protocol follows from the paper of Camenisch and Lysyanskaya ([7]) and then, the same security arguments can be applied.

3 A New Electronic Voting Scheme

We now present our voting scheme, based on list signatures introduced in previous section.

3.1 Electronic Voting

An electronic voting scheme is a set of protocols allowing all legitimate voters (and only them) to vote and a set of authorities to collect votes and calculate the result. There are several required security properties for such schemes. formally speaking:

Definition 3. *The following properties must be satisfied by a secure electronic voting scheme:*
(i) Democracy: only registered voters are allowed to vote and no voter can vote twice.
(ii) Anonymity: all votes must be secret.
(iii) No Partial Results: it is impossible to perform partial tabulation before the end of the

poll.
(iv) Public Verifiability: any party can check that ballots are correctly cast, that only invalid ballots are discarded, and that the outcome of the election is consistent with the valid cast ballots.
(v) Receipt-freeness: the voter must not be able to prove to a third party the way he voted.

3.2 Preliminaries

3.2.1 Overview

That our proposal uses a list signature scheme allows us to directly achieve the two first security requirements (democracy and anonymity). In addition, we use an anonymous channel (a channel in which it is impossible to link the sender of the message and the message itself, e.g. a cybercafe or a mix-nets) so as to achieve full anonymity of voters.
To meet the third security requirement, each voter encrypts his vote with a public key corresponding to a private one only known by one authority (or shared by several authorities (see Sect. 3.2.2)). If we trust this authority (or at least one of the authorities), the votes will not be decrypted until the end of the ballot.
Various entities participate to the electronic voting scheme: the Authority (the entity who is in charge of the election by generating all necessary data for the good execution of the protocols), the Voter V_i (a person who is allowed to vote) and the Group of scrutineers (the set of entities involved in the cryptosystem (see Sect. 3.2.2)). Finally, a public database (the bulletin board) will collect all ballot papers.

3.2.2 Cryptosystem

Our solution needs a cryptosystem which is semantically secure (the ciphertext discloses no partial information about its plaintext beyond the length of that plaintext). Moreover, if it is possible to trust a single authority, we can then use a classical cryptosystem (e.g., the El Gamal cryptosystem which is known to be semantically secure). But if this assumption cannot be made, it is desirable to use a threshold decryption.
A (t, n) threshold scheme ($t \leq n$) is a method by which n users obtain a secret share from an initial secret. Any t or more users who put in common their shares can easily recover the initial share but any group knowing only $t-1$ or fewer shares cannot. Well-known solutions are based either on a modification of the El Gamal encryption scheme or on the Paillier cryptosystem (see [3]).

3.3 Description

3.3.1 Registration of Voters

Our electronic voting scheme has several phases. During the first one, each voter executes the Join protocol with TA in order to obtain the capacity of making a list signature: he is now registered in the electoral list and can participate in all future elections. An example of list signature scheme is showed in Sect. 2 and, for the rest of this paper, we will denote by $Sig_{Liste,V_i}(M)$ the list signature of the message M by the user V_i. If a voter is no longer authorized to vote after the last election (this can occur for various reasons), the first authority can delete his/her right of signing by using the revocation algorithm (see Sect. 2).

3.3.2 Preparation of the Election

Before an election, the first authority (or another one) prepares it by choosing sequence parameters. This authority runs the "organization of a sequence" algorithm (see Sect. 2.1 and 2.2.4). Then, all valid voters can vote in this election (and, consequently, can vote only once). The group of scrutineers publishes an encryption public key and each one keeps secret his share of the decryption private key. We will denote by E_{PK_e} the encryption algorithm (using the public key PK_e) and D_{SK_e} the decryption algorithm used by scrutineers who share the private key SK_e.

3.3.3 Ballot Progress

During the ballot progress, the voter V_i first makes his choice v_i. He then performs the following:

- Compute $s_i = Sig_{Liste,V_i}(v_i)$.
- Compute $d_i = E_{PK_e}(v_i, s_i)$.
- Put (d_i) into the database through an anonymous channel (e.g. mix-nets or a cybercafe).

3.3.4 Counting of Votes

At the end of the ballot, the database contains all ballots of all voters (who have participated in this election). The group of scrutineers can then collect all these ballots and start counting the votes. They use the distributed decryption algorithm and the verification algorithm of the list signature scheme (see Sect. 2.2.6 for details) and do the following for each entry of the database.

- Take the next entry of the database (d_i)
- Obtain $(s_i, v_i) = D_{SK_e}(d_i)$.
- Verify the list signature scheme s_i with the message v_i.
- Test if this voter has not yet voted (which is possible due to the property of partial linkability of the list signature scheme[2]).
- Put (d_i, s_i, v_i) in the database.
- Modify the result of the election in accordance with v_i.

At the end of the counting, the group of scrutineers publishes their private key SK_e so that everybody can make the same counting as they have done.

3.4 Security Properties

We briefly give strong evidence that our scheme verifies all properties introduced in Sect. 3.1. (i) Democracy: this property is verified as only allowed voters can obtain a valid certificate to make a list signature of their ballot. TA knows to which it has provided a member certificate and then would not do it twice for a same person. Furthermore, we have seen that neither TA nor RA can create fake voters (see Sect. 2.2.7). Finally, as the list signature scheme is secure, and since each election takes place within a single sequence of a list signature scheme,

[2] Scrutineers have to make a comparison between the current signature (and more particularly T_4, see Sect. 2.2.6) and ancient ones. This type of comparison is inherent in all known electronic voting systems.

each voter can vote only once. In fact, if a valid voter tries to vote twice, scrutineers could notice it (see Sect. 2.2.6).[3]
(ii) Anonymity: this property is achieved since the list signature is anonymous and partially unlinkable and since the ballot is sent through an anonymous channel. Furthermore, the encrypted ballot paper reveals nothing about the sender as the public key of the cryptosystem is the same for all voters.
(iii) No Partial Results: there cannot be partial results under the hypothesis that at least one member of the group of scrutineers is honest. In this case, this one would not want to decrypt votes before the end of the ballot and as our proposal use a threshold cryptosystem, the presence of all scrutineers is required. Furthermore, the semantic security of the public key cryptosystem ensures that nothing is revealed concerning the vote v_i.
(iv) Public Verifiability: follows from the fact that the decryption private key will be revealed and that everybody (even a simple observer) can follow the evolution of the database (and then notice a non valid modification of it). Consequently, at the end of the counting of votes, everybody can do the same work as scrutineers have done.
(v) Receipt-freeness: with our solution, each voter can prove for whom he had voted. This seems to be inherent in case of on-line voting but can be solved by using a smart card in case of off-line voting (see [4] for example).
Moreover, each voter can prove that he participated in the election by proving that he produced one entry of the database (without revealing which one). This proof will not be presented in this paper and is based on the proof of OR introduced by Cramer et al. (see [11] and [17] for details).

4 Conclusion

In this paper, we have presented a new type of signature scheme, a variant of group signature, called list signature. We have proposed an example of such a scheme and shown that it verifies all required security properties. In particular, signatures produced in this scheme are provably coalition-resistant and existentially unforgeable under the Strong-RSA Assumption. After that, we have proposed an electronic voting scheme that makes use of this new concept. This scheme, a simple combination of a list signature scheme and of a (threshold) semantically secure cryptosystem, allows to meet very easily the fundamental requirements of security in electronic voting.

Acknowledgments

We are very grateful to Marc Girault for numerous discussions and comments. We would also like to thank the anonymous referees for their useful remarks.

References

[1] G. Ateniese, J. Camenisch, M. Joye, G. Tsudik. A Practical and Provably Secure Coalition-Resistant Group Signature Scheme. Crypto'2000, volume 1880 of LNCS, pages

[3] There are many ways of reacting here. Scrutineers can reject all votes, or they can only keep the last one (or the first one), each vote being dated by the voter. In the latter case, it can viewed as a possibility for each voter to modify his decision (as for a paper submission).

255-270. Springer-Verlag, 2000.

[2] N. Barić, B. Pfitzmann. Collision-Free Accumulators and Fail-Stop Signature Schemes Without Trees. Eurocrypt'97, volume 1233 of LNCS, pages 480-484. Springer-Verlag, 1997.

[3] O. Baudron, P.-A. Fouque, D. Pointcheval, G. Poupard, J. Stern. Practical Multi-Candidate Election System. PODC'01. ACM, 2001.

[4] J. Benaloh-Cohen, D. Tuinstra. Receipt-Free Secret-Ballot Elections. 26th ACM Symposium on the Theory of Computing (STOC'94), pages 544-553. ACM Press, 1994.

[5] F. Boudot, J. Traoré. Efficient Publicly Verifiable Secret Sharing Schemes with Fast or Delayed Recovery. ICICS'99, volume 1726 of LNCS, pages 87-102. Springer-Verlag, 1999.

[6] J. Camenisch, A. Lysyanskaya. An Efficient System for Non-transferable Anonymous Credentials with Optional Anonymity Revocation. Eurocrypt 2001, volume 2045 of LNCS, pages 93-118. Springer-Verlag, 2001.

[7] J. Camenisch, A. Lysyanskaya. Efficient Revocation of Anonymous Group Membership Certificates and Anonymous Credentials. Crypto'2002, volume 2442 of LNCS, pages 61-76. Springer-Verlag, 2002.

[8] D. Chaum. Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. Communications of the ACM. 1981.

[9] D. Chaum. Blind Signatures for Untraceable Payments. Crypto'83, LNCS, page 153. Springer-Verlag, 1984.

[10] D. Chaum, E. van Heyst. Group Signatures. Eurocrypt'91, volume 547 of LNCS, pages 257-265. Springer-Verlag, 1991.

[11] R. Cramer, I.B. Damgård, B. Schoenmakers. Proof of Partial Knowledge and Simplified Design of Witness Hiding Protocols. Crypto'94, volume 839 of LNCS. Springer-Verlag, 1994.

[12] A. Fiat, A. Shamir. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. Crypto'86, volume 263 of LNCS, pages 186-194. Springer-Verlag, 1987.

[13] E. Fujisaki, T. Okamoto. Statistical Zero-Knowledge Protocols Solution to Identification and Signature Problems. Crypto'97, volume 1294 of LNCS, pages 16-30. Springer-Verlag, 1997.

[14] A. Fujioka, T. Okamoto, K. Ohta. A practical secret voting scheme for large scale elections. Auscrypt92, volume 718 of LNCS, pages 248-259. Springer-Verlag, 1992.

[15] T. Nakanishi, T. Fujiwara, H. Watanabe. A Linkable Group Signature and Its Application to Secret Voting. Trans. IPS. Japan, Vol.40, No.7, pp.3085-3096, July 1999.

[16] D. Pointcheval, J. Stern. Security proofs for signatures schemes. Eurocrypt'96, volume 1070 of LNCS, pages 387-398. Springer-Verlag, 1996.

[17] B. Schoenmakers. Efficient Proofs of OR. manuscript, 1993.