An electronic voting scheme

Tatsuaki Okamoto
NTT Laboratories
1-2356 Take, Yokosuka-shi, 238-03 Japan
email: okamoto@sucaba.isl.ntt.jp

Abstract

This paper proposes a practical large scaled secret voting scheme which satisfies the receipt-free property* as well as the fairness property†. This scheme uses three realistic tools: a non-anonymous channel, a bulletin board, and a private channel. The participants of the scheme are the voters and voting commissioners (administrators, privacy commission members, and timeliness commission members). There are four stages to this scheme: the authorizing stage, voting stage, claiming stage, and counting stage. A voter performs just one round message exchange with the administrators in the authorizing stage, and is required to send just one message in the voting stage.

Keywords

Secure Communication, Electronic Voting, Electronic Ballot, Electronic Questionnaire, Auction, Cryptology,

1 INTRODUCTION

Many electronic secret voting schemes have been proposed in the last ten years. We can classify these electronic secret voting schemes into three types: one is the multi-party protocols (or secure computation protocols) [GMW87, BGW88, CCD88], another is the schemes that use the higher degree residue encryption (or homomorphic encryption) technique [BY86, BT94, CF85, Ive92, SK94], and the other is the schemes based on the blind signature and an anonymous channel [Cha88, FOO92, Oht88, SK95].

Here note that the first type requires no party other than the voters, while the second and third types require additional participants, e.g., a center or an administrator.

In the first type, or the schemes that use the multi-party protocol [GMW87, BGW88, CCD88], all procedures are managed by just the voters, however, many communication

*With the receipt-free property, there is no receipt of a vote, which could be used by another party to buy a vote or to coerce a voter.

†With the fairness property, no one can know even intermediate voting results

acts are required among voters. Therefore, this approach is very important theoretically, but is impractical, especially for large scale voting.

The second type, or the schemes that use higher degree residue encryption (or homomorphic encryption) [BY86, CF85, CFSY96, Ive92, SK94], needs centers to protect voter privacy. However, these schemes are still rather impractical for large scale elections, since a lot of communication and computation overhead is needed to prevent the dishonesty of voters by using zero-knowledge (like) protocols. Moreover, the schemes are limited to only yes/no voting. Although they can be extended to multiple value voting, the complexity increases a lot [BY86, CFSY96].

The third type is the most suitable and practical for large scale elections, since the communication and computation overhead is fairly small even if the number of voters is large, and several practical schemes [Cha88, FOO92, Oht88, SK95] of this type have been proposed recently. Moreover, this type of schemes naturally realize the multiple value voting. Such schemes contain three stages: the authorizing stage, voting stage, and counting stage. In the authorizing stage, the administrators issue blind signatures, in the voting stage, the voters send their votes with the administrator's signatures to the counter through anonymous channels, and in the counting stage, the votes are verified and counted.

Major problems to be solved in the schemes of the third type are fairness, privacy, anonymity, and receipt-freeness. The fairness problem is that when the center or a third party can know intermediate voting results by opening the ballots, he can affect the final voting result by leaking the intermediate results. The privacy problem is that a voter's privacy can be violated, especially when the voter notices that his vote was not counted correctly and claims it by showing his vote after the counting stage. The anonymity problem is that it is hard to realize an anonymous channel within a real network, since usual digital channels such as ISDN and email networks are not anonymous channels. The receipt-freeness problem is that the receipt of a vote, which proves how a voter voted, could be used by another party to buy a vote or to coerce a voter [BT94, JSI96, SK95].

The originally proposed schemes of this type [Cha88, Oht88] did not solve these problems. Fujioka et al. proposed a scheme which overcomes the fairness and privacy problems [FOO92]; however, messages must be sent to the center twice to guarantee fairness. Moreover, it does not overcome the anonymity and receipt-free problems. Sako and Kilian [SK95] proposed a receipt-free voting scheme, but the computational complexity is quite large when it is large scaled, and the voting is limited to yes/no.

This paper proposes a practical large scale secret voting scheme (of the third type), which solves the fairness, privacy, anonymity, and receipt-freeness.

The correctness of the voting result in the proposed scheme is guaranteed even if some of the administrators and all commission members conspire, under the condition that each voter checks his/her individual information on a board. Voter privacy is ensured unless all the privacy commission members conspire simultaneously. The scheme also realizes voting fairness unless all the timeliness commission members conspire simultaneously. If private channels exist, the receipt-freeness is satisfied unless all timeliness commission members conspire simultaneously.

2 PARTICIPANTS OF THE PROPOSED SCHEME

The participants of the scheme are voters, V_i ($i = 1, 2, \dots, I$), and voting commissioners, which consists of administrators, A_j ($j = 1, 2, \dots, J$), privacy commission members, P_k ($k = 1, 2, \dots, K$), and timeliness commission members, T_l ($l = 1, 2, \dots, L$).

3 BASIC FRAMEWORK OF THE PROPOSED SCHEME

This section introduces the basic framework of our voting scheme. For simplicity of description, here, we assume that the numbers of administrators, and timeliness commission member are one, and so we simply write A and T for the administrator and timeliness commission (member). In addition, we do not use privacy commission members in this basic framework, and instead use anonymous channels. Later we will show how to replace anonymous channels by the Mixnet technique with privacy commission members.

There are four stages in this framework: the authorizing stage, voting stage, claiming stage, and counting stage.

3.1 Authorizing stage

In this stage, each voter obtains the signature of the administrator to his ballot in a blind signature manner. Here, the RSA blind signature is used, but we can use any other blind signatures [OO89, Oka92, Sta96].

1. Voter V_i makes his/her vote v_i and computes

$$m_i = BC(v_i \| r_i),$$

using a random number r_i . Here, BC is a bit-commitment function [CFSY96]. V_i computes

$$x_i = H(m_i) t_i^e \bmod n,$$

where t_i is a random number in Z_n , (e, n) is the RSA public key of A for signatures, and H is a hash function. V_i generates his/her signature

$$z_i = S_{V_i}(x_i)$$

for x_i . V_i also computes

$$E_A(x_i \| z_i \| ID_{V_i}),$$

where E_A is a non-malleable public-key encryption [BR94] using A 's public-key, and $\|$ denotes concatenation.

2. V_i sends $E_A(x_i \| z_i \| ID_{V_i})$ to A .
3. A decrypts the message, and checks that voter V_i has the right to vote. If V_i doesn't have the right, A rejects. A also checks whether or not V_i has already applied for a signature. If V_i has already applied, A rejects. If V_i is accepted, A checks the signature z_i of message x_i . If they are valid, then A generates signature

$$y_i = x_i^{1/e} \bmod n.$$

A sends y_i to V_i .

4. V_i checks the signature y_i of message x_i . If it is valid, voter V_i retrieves A 's signature $s_i = H(m_i)^{1/e} \bmod n$ of message m_i by $s_i = y_i/t_i \bmod n$.

3.2 Voting stage

V_i sends (m_i, s_i) to the bulletin board, and sends (v_i, r_i, m_i) to timeliness commission member T through anonymous channels.

3.3 Claiming stage

After the voting stage, a fixed term (e.g. three hours) is set for the claiming stage, during which a voter can make a claim that his/her vote has been neglected or that a forged vote has been registered in the list.

In the claiming stage, V_i checks that his/her ballot is listed on the bulletin board (ballot list). If his/her vote is not listed, then V_i claims this by showing (m_i, s_i) .

3.4 Counting stage

After the claiming stage, the counting stage starts.

1. T sends (v_i, r_i, m_i) ($i=1,2,\dots,I$) to the bulletin board. If m_i is claimed to be invalid in the claiming stage, then T sends no message regarding m_i .
2. Anyone can check whether

$$m_i = BC(v_i, r_i).$$

They then count v_i if it is correct. If the check regarding m_i is not correct, they send to the board ("invalid", m_i).

3. Anyone can obtain the same final result by counting the votes.

4 RECEIPT-FREE VERSION OF THE BASIC FRAMEWORK

The above-mentioned basic framework does not satisfy the receipt-free condition, since (v_i, r_i) is published and specific to each voter V_i , therefore it can be a receipt (transferable proof) of V_i 's voting v_i .

This section introduces the receipt-free version of the basic framework. In this version, V_i can confirm whether his/her vote v_i is counted or not, but, at the same time, can make various receipts which can be used to cheat vote buyers and coercers. Here the following two tools play the essential role: trap-door bit-commitments [JSI96], and zero-knowledge proof of proving the correctness of the vote, v_i , while concealing the relationship between m_i and v_i .

4.1 Authorizing stage

We now assume that several parameters, p, q, g, h , are generated and published by the system, where p and q are prime, $q|p-1$, g and h are in Z_p^* , and $\text{order}(g) = \text{order}(h)$. (i.e., $g^q \equiv h^q \equiv 1 \pmod{p}$, and $g \neq h \neq 1$.) Here, α such that $h = g^\alpha \bmod p$ is not known

to any party. For example, g and h are generated using a publicly known pseudo-random generation algorithm and exponentiation to adjust the order.

1. V_i randomly generates $\alpha_i \in Z_q$, and calculates $G_i = g^{\alpha_i} \bmod p$. We then define $BC(v_i, r_i) = g^{v_i} G_i^{r_i} \bmod p$. Here, note that $BC(v_i, r_i)$ is a trap-door bit-commitment, since V_i can open this bit-commitment in many ways, (v_i, r_i) , (v'_i, r'_i) , etc., using α_i such that $v_i + \alpha_i r_i \equiv v'_i + \alpha_i r'_i \pmod{q}$.

V_i makes his/her vote v_i and computes

$$m_i = BC(v_i, r_i) = g^{v_i} G_i^{r_i} \bmod p,$$

using a random number r_i . V_i computes

$$x_i = H(m_i \| G_i) t_i^e \bmod n,$$

where t_i is a random number in Z_n , and (e, n) is the RSA public key of A for signatures. V_i generates his/her signature

$$z_i = S_{V_i}(x_i)$$

for x_i . V_i also computes

$$E_A(x_i \| z_i \| ID_{V_i}).$$

2. The remaining procedures of the authorizing stage are the same as those in the basic framework. Finally, V_i obtains A 's signature $s_i = H(m_i \| G_i)^{1/e} \bmod n$ of message $(m_i \| G_i)$.

4.2 Voting stage

V_i sends $(m_i \| G_i, s_i)$ to the bulletin board through anonymous channels. V_i also sends (v_i, r_i, m_i) to timeliness commission member T through anonymous and private channels.

4.3 Claiming stage

1. V_i checks that his/her ballot is listed on the bulletin board (ballot list). If his/her vote is not listed, then V_i claims this by showing $(m_i \| G_i, s_i)$.
2. T checks whether (v_i, r_i) is a correct open value of m_i . If it is not correct, T claims it on the board. The corresponding voter asks the system to investigate the cause, or sends the correct open value to T . T then shows the result on the board.

4.4 Counting stage

In the receipt-free version, the counting stage is completely different from that in the basic framework. The open value, (v_i, r_i) , of m_i is directly shown on the board in the basic framework. Therefore, the linkage between m_i and v_i is published. In contrast, in the receipt-free version, only v_i is published on the board, and a zero-knowledge (like) proof is given by T to prove that v_i is a correct open value of a member of a group of m_i 's. Here note that the linkage between m_i and v_i is concealed, but v_i is guaranteed to be an authorized vote in the authorization stage.

1. T makes many disjoint groups of votes, where any vote belongs to a group, and each group consists of several votes which cover all possible votes (candidates). Here, we explain the procedure using the example of a group consisting of 5 votes (say v_1, v_2, v_3, v_4, v_5). T generates a random permutation $\pi \in S_5$, and T shows $(m_1, m_2, m_3, m_4, m_5)$ and $(v'_1, v'_2, v'_3, v'_4, v'_5) = (v_{\pi(1)}, v_{\pi(2)}, v_{\pi(3)}, v_{\pi(4)}, v_{\pi(5)})$ on the board. T then gives a non-interactive proof, σ , that $\{v'_1, v'_2, v'_3, v'_4, v'_5\} = \{v_1, v_2, v_3, v_4, v_5\}$ (equivalence as a set) without giving any linkage between m_i and v_i .
For example, $\pi : (1, 2, 3, 4, 5) \rightarrow (2, 4, 1, 5, 3)$, then $(v'_1, v'_2, v'_3, v'_4, v'_5) = (v_2, v_4, v_1, v_5, v_3)$.
Note: In the basic framework, if m_i is claimed to be invalid in the claiming stage, then T does not give open value of m_i to protect the privacy of the claiming voter. However, in the receipt-free version, the vote of a claiming voter can be counted.
2. Anyone can check whether σ of each group is correct or not. They then count v_i if it is correct. If the check regarding a group is not correct, they send to the board ("invalid", σ of a group).
3. Anyone can obtain the same final result by counting the votes.

Zero-knowledge interactive proof (basic version of σ)

Input: $(m_1, m_2, m_3, m_4, m_5)$ and $(v'_1, v'_2, v'_3, v'_4, v'_5)$.

What to prove: T (prover) knows π and r_i ($i = 1, \dots, 5$) such that $m_i = BC(v_i, r_i)$ and $v'_i = v_{\pi(i)}$.

Repeat the following procedure $k(= \text{poly}(|p|))$ times.

1. T (prover) generates random permutation $\delta \in S_5$. T also randomly generates $u_i, s_i, t_i \in Z_q$ ($i = 1, \dots, 5$).

$$Z_i = m_i G_i^{s_i} h^{u_i} \bmod p,$$

$$W_i = g^{v'_{\delta(i)}} h^{t_i} \bmod p,$$

where $i = 1, \dots, 5$.

T sends $X = (Z_1, \dots, Z_5, W_1, \dots, W_5)$ to Verifier.

2. Verifier randomly selects a bit $e \in \{0, 1\}$, and sends it to T
3. If $e = 0$, then T sends $Y = (\delta, u_i, s_i, t_i$ ($i = 1, \dots, 5$)) to Verifier. If $e = 1$, T calculates

$$\rho = \pi^{-1} \circ \delta^{-1},$$

$$x_i = r_i + s_i \bmod q,$$

$$y_i = u_i - t_{\rho(i)} \bmod q,$$

and sends $Y = (\rho, x_i, y_i$ ($i = 1, \dots, 5$)) to Verifier.

4. If $e = 0$, then Verifier checks whether the following hold or not:

$$Z_i = m_i G_i^{s_i} h^{u_i} \bmod p,$$

$$W_i = g^{v'_{\delta(i)}} h^{t_i} \bmod p.$$

If $e = 1$, Verifier checks whether the following holds or not:

$$W_{\rho(i)} G_i^{x_i} h^{y_i} \equiv Z_i \pmod{p}.$$

Non-interactive proof: σ

Now we will show a way to make a non-interactive proof, σ , based on the the above-mentioned zero-knowledge interactive proof. Here we assume a random function, $F : \{0, 1\}^* \rightarrow \{0, 1\}^k$. In practice, F can be replaced by a realistic one-way hash function such as SHA and MD5 [BR94].

T generates k times $X = (Z_1, \dots, Z_5, W_1, \dots, W_5)$ in the same manner as the above-mentioned zero-knowledge interactive proof (say $X_j = (Z_1, \dots, Z_5, W_1, \dots, W_5)_j$ ($j = 1, \dots, k$)). T then generates

$$(e_1, e_2, \dots, e_k) = F(X_1, \dots, X_k).$$

T also generates T 's second message Y_j of the zero-knowledge interactive proof, corresponding to X_j and e_j ($j = 1, \dots, k$). So,

$$\sigma = (X_1, \dots, X_k, e_1, \dots, e_k, Y_1, \dots, Y_k).$$

5 MULTIPLE ADMINISTRATORS AND TIMELINESS COMMISSION MEMBERS

The above-mentioned schemes use single administrator and single timeliness commission member. In this section, we show how to introduce multiple administrators and multiple timeliness commission members to improve the security. Here, we assume that there are J administrators, A_j ($j = 1, 2, \dots, J$), and L timeliness commission members, T_l ($l = 1, 2, \dots, L$).

5.1 Authorizing stage

In the authorizing stage, voter V_i executes the above-mentioned single-administrator authorizing procedure for each administrator A_j , where $m_i \| G_i$ is common among all procedures with $A_1, \dots, A_J$. Finally, V_i obtains administrators' signatures of m_i ,

$$(s_{i1}, \dots, s_{iJ}) = (S_{A_1}(m_i \| G_i), \dots, S_{A_J}(m_i \| G_i)).$$

5.2 Voting stage

Here, V_i splits v_i and r_i into L parts as follows:

$$v_i = v_{i1} + \dots + v_{iL} \bmod q,$$

$$r_i = r_{i1} + \dots + r_{iL} \bmod q.$$

Then, V_i sends $(m_i \| G_i)$ along with $(s_{i1}, \dots, s_{iJ})$ to the bulletin board through anonymous channels. V_i also sends (v_{il}, r_{il}, m_i) to timeliness commission members T_l ($l = 1, 2, \dots, L$) through anonymous and private channels.

5.3 Claiming stage

1. V_i checks that his/her ballot is listed on the bulletin board (ballot list). If V_i 's vote is not listed, then V_i claims this by showing $(m_i \| G_i)$ and $(s_{i1}, \dots, s_{iJ})$.

2. T_l calculates

$$m_{il} = g^{v_{il}} G_i^{r_{il}} \bmod p,$$

and sends m_{il} to a private board which is read by only timeliness committee members. T_l checks

$$m_i = m_{i1} \cdots m_{iL} \bmod p.$$

If it is not correct, T_l claims it on the board. The corresponding voter asks the system to investigate the cause, or sends the correct values, (v_{il}, r_{il}, m_i) to T . T then shows the result on the board.

5.4 Counting stage

In this stage, all members of T_l collaborate to generate v_i and σ while keeping r_l secret to other members.

First, T_l sends each v_{il} to their private board and calculate

$$v_i = v_{i1} + \cdots + v_{iL} \bmod q.$$

Next they collaborate to make σ .

1. First, they agree with the grouping and share random permutations, π and δ . They generate k pairs of permutations of (π, δ) for k round parallel executions of the basic zero-knowledge protocol. (Here, we also assume the number of votes in the group is 5.)
2. Then T_l ($l = 1, \dots, L$) generates randomly $u_{il}, s_{il}, t_{il} \in Z_q$ ($i = 1, \dots, 5$), and computes

$$Z_{il} = m_{il} G_i^{s_{il}} h^{u_{il}} \bmod p,$$

$$W_{il} = g^{v_{\pi(i)l}} h^{t_{il}} \bmod p,$$

where $i = 1, \dots, 5$, and $v'_{il} = v_{\pi(i)l}$. T_l repeats the procedure k times and obtains $X_{jl} = (Z_{1l}, \dots, Z_{5l}, W_{1l}, \dots, W_{5l})_j$ ($j = 1, \dots, k$). T_l sends X_{jl} to their private board.

3. They then obtain $X_j = (Z_1, \dots, Z_5, W_1, \dots, W_5)_j$ by calculating $Z_i = \prod_{l=1, \dots, L} Z_{il} \bmod p$, and $W_i = \prod_{l=1, \dots, L} W_{il} \bmod p$ ($i = 1, \dots, 5$).
4. They calculate

$$(e_1, e_2, \dots, e_k) = F(X_1, \dots, X_k).$$

5. If $e_j = 0$, T_l sends $Y_l = \{(u_{il}, s_{il}, t_{il}) \mid i = 1, \dots, 5\}$ to their board. If $e_j = 1$, T_l calculates

$$x_{il} = r_{il} + s_{il} \bmod q,$$

$$y_{il} = u_{il} - t_{\rho(i)l} \bmod q,$$

and sends $Y_l = \{(x_{il}, y_{il}) \mid i = 1, \dots, 5\}$ to their board. where $\rho = \pi^{-1} \circ \delta^{-1}$.

They then obtain $Y_j = (\delta, u_i, s_i, t_i \mid i = 1, \dots, 5)_j$ by calculating $u_i = \sum_{l=1, \dots, L} u_{il} \bmod q$, $s_i = \sum_{l=1, \dots, L} s_{il} \bmod q$, and $t_i = \sum_{l=1, \dots, L} t_{il} \bmod q$, when $e_j = 0$.

When $e_j = 1$, they obtain $Y_j = (\rho, x_i, y_i \mid i = 1, \dots, 5)_j$ by calculating $x_i = \sum_{l=1, \dots, L} x_{il} \bmod q$, and $y_i = \sum_{l=1, \dots, L} y_{il} \bmod q$.

6. Finally, they send

$$\sigma = (X_1, \dots, X_k, e_1, \dots, e_k, Y_1, \dots, Y_k)$$

to the public board.

6 USING MIXNET

In the above-mentioned schemes, we use anonymous channels. Here we introduce the standard Mixnet technique to replace the anonymous channels by non-anonymous channels, and privacy commission members execute the Mixnet procedures. If at least one of the members is not corrupted by the adversary, anonymity is guaranteed. Here, we omit a detailed description of the procedure due to the space limitations (see [Cha81, SK94, Sta96] for Mixnet).

By introducing Mixnet technique, the proposed voting scheme requires three realistic tools: a non-anonymous channel (in place of an anonymous channel), a bulletin board, and a private channel.

7 EFFICIENCY

The heaviest part of our voting scheme in terms of computational complexity is computing σ . Here, we estimate the efficiency of computing σ .

First, the computational complexity of σ is linear in the number of voters. For each vote, $2k$ modular exponentiations with double bases are the majority part for each T_i . If we assume $|p| = 1024$, and $|q| = 160$, then one modular exponentiation requires 160×1.75 modular multiplications with 1024 bits on average, by using the standard binary method for double bases. Therefore, the computational complexity for σ is around $560k$ modular multiplications per voter for each T_i .

Here note that this computation can be executed immediately after T_i obtains (v_{ii}, r_{ii}) . In other words, even if the grouping for σ is not determined, such computation for each vote can be pre-computed. Then, the amount of T_i 's pre-computation on receiving (v_{ii}, r_{ii}) , around $560k$ modular multiplications, is practical, since it is less than k RSA decryptions with 1024 bit modulus. (e.g., $k = 128$.)

ACKNOWLEDGMENTS

I would like to thank Phil Rogaway for many invaluable suggestions and comments on the preliminary version of this paper.

REFERENCES

- [BGW88] M. Ben-Or, S. Goldwasser, and A. Wigderson, "Completeness Theorems for Non-Cryptographic Fault-Tolerant Distributed Computation", Proc. of STOC'88, pp.1-10 (1988).
- [BR94] M. Bellare and P. Rogaway, "Optimal Asymmetric Encryption", Proc. of Eurocrypt'94, LNCS 950, Springer-Verlag (1994).
- [BT94] J. Benaloh and D. Tuinstra, "Receipt-Free Secret-Ballot Elections", Proc. of STOC'94, pp.544-553 (1994).

- [BY86] J. Benaloh and M. Yung, "Distributing the Power of a Government to Enhance the Privacy of Votes", Proc. of PODC'86, pp.52–62 (1986).
- [Cha81] D. Chaum, "Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms", Communications of the ACM, Vol.24, No.2, pp.84–88 (1981).
- [Cha85] D. Chaum, "Security without Identification: Transaction systems to Make Big Brother Obsolete", Communications of the ACM, Vol.28, No.10, pp.1030–1044 (1985).
- [Cha88] D. Chaum, "Elections with Unconditionally-Secret Ballots and Disruption Equivalent to Breaking RSA", Proceedings of Eurocrypt'88, LNCS 330, Springer-Verlag, pp.177–182 (1988).
- [CCD88] D. Chaum, C. Crépeau, and I. Damgård, "Multiparty Unconditionally Secure Protocols", Proc. of STOC'88, pp.11–19 (1988).
- [CF85] J. Cohen and M. Fisher, "A Robust and Verifiable Cryptographically Secure Election Scheme", Proc. of FOCS, pp.372–382 (1985).
- [CFSY96] R. Cramer, M. Franklin, B. Schoenmakers, and M. Yung, "Multi-Authority Secret-Ballot Elections with Linear Work", Proc. of Eurocrypt'96, LNCS 1070, Springer-Verlag, pp.72–82 (1996).
- [FOO92] A. Fujioka, T. Okamoto, and K. Ohta, "A Practical Secret Voting Scheme for Large Scale Elections", Proc. of Auscrypt '92, LNCS, Springer-Verlag, pp. 244–251 (1992).
- [GMW87] O. Goldreich, S. Micali, and A. Wigderson, "How to Play Any Mental Game, or a Completeness Theorem for Protocols with Honest Majority", Proc. of STOC, pp.218–229 (1987).
- [Ive92] K. R. Iversen, "A Cryptographic Scheme for Computerized General Elections", Proc. of Crypto '91, LNCS 576, Springer-Verlag, pp.405–419 (1992).
- [JSI96] M. Jakobsson, K. Sako, and R. Impagliazzo, "Designated Verifier Proofs and Their Applications", Proc. of Eurocrypt '96, LNCS 1070, Springer-Verlag, pp.143–154 (1996).
- [MR91] S. Micali, and P. Rogaway, "Secure Computation (Chapter 1-3)", Manuscript (August 9, 1991). (The abstract version: Advances in Cryptology — CRYPTO '91, Lecture Notes in Computer Science 576, Springer-Verlag, Berlin, pp.392–404 (1992)).
- [Oht88] K. Ohta, "An Electrical Voting Scheme using a Single Administrator", 1988 Spring National Convention Record, IEICE, Japan, A-294 (1988).
- [Oka92] T. Okamoto, "Provably Secure and Practical Identification Schemes and Corresponding Signature Schemes", Proc. of Crypto'92, LNCS 740, Springer-Verlag, pp.31–53 (1993).
- [OO89] T. Okamoto, and K. Ohta, "Divertible Zero-Knowledge Interactive Proofs and Commutative Random Self-Reducible," Proc. of Eurocrypt '89, LNCS 434, Springer-Verlag, pp.134–149 (1990).
- [SK94] K. Sako, and J. Kilian, "Secure Voting Using Partially Compatible Homomorphisms", Proc. of Crypto'94, LNCS 839, Springer-Verlag, pp.411–424 (1994).
- [SK95] K. Sako, and J. Kilian, "Receipt-Free Mix-type Voting Scheme", Proc. of Eurocrypt'95, LNCS 921, Springer-Verlag, pp.393–403 (1995).
- [Sta96] M. Stadler, "Cryptographic Protocols for Revocable Privacy", PhD Thesis, ETH (1996), <http://www.inf.ethz.ch/personal/stadler/thesis.ps>.