SCIS2000 The 2000 Symposium on Cryptography and Information Security
Okinawa, Japan, January 26-28, 2000

An Electronic Voting Scheme with Revocable Threshold Blind Signatures

追跡可能ブラインド署名に基づく電子投票

Masayuki Abe, Miyako Ohkubo, Atsushi Fujioka, Fumitaka Hoshino *

Abstract— In this paper, we construct an electronic voting scheme by using a simple (i.e. not necessarily verifiable) Mix-net and *revocable threshold blind signatures* in order to eliminate the use of physical anonymous channel. An adversary that can obtain views from up to less than half of mix-servers and can control at most $n-2$ of n voters is tolerable.

Keywords: Electronic Voting, Blind signatures, Revocable Blind Signatures, Mix-net

1 Introduction

Background. Secret electronic voting is an application of multi-party protocol where a large number of studies has been done so far. In particular, schemes based on blind signatures are considered as practical ones because 1) they are so flexible that any voting policy can live with, 2) efficient in computation and communication, 3) conceptually easy to understand. Indeed, some implementation and field experiments are reported [12].

A big obstacle of using blind-signature-based voting schemes in a real use is the use of anonymous channels. Currently widely available networks such as Internet do not provide sufficient anonymity. We do not consider using a public voting booth or public telephone infrastructure to achieve anonymity because they definitely limit the availability.

A possible alternative would be to use a Mix-net [7, 17], though some classical construction do not cover all the necessary properties physical anonymous channels have. More reliable system can be built by using a robust verifiable Mix-net [15, 1, 13, 14, 2]. However, they are computationally expensive in general [1].

Problem. Suppose that a simple Mix-net, which does not guarantee one-to-one correspondence between the input ciphertexts and output messages, is used as an anonymous channel. Also suppose that we have an invalid signature in the resulted list. We may trace the mix-net backward in order to identify the voter who sent the invalid signature. But it is not enough. Because there could be a mix-server, say the last one for simplicity, colluding with the voter, which has replaced a correct signature sent from another voter with a correct signature issued to the bad voter. Thus, we can not either correct the result or detect the colluding mix-server unless the *bad* voter is so cooperative to present valid signature issued to him, or each voter checks whether his vote is counted.

The same problem arises if we allow voters to abstain after getting blind signatures. Even worse, in this case, we can not accuse the suspicious voter because of his abstention.

Our Contribution. This paper presents a practical blind-signature-based voting scheme that uses a simple Mix-net instead of physical anonymous channels or computationally expensive verifiable Mix-nets. As a building block, we construct a revocable threshold blind signature scheme with which blind signatures are issued iff a quorum of administrators cooperate and blindness of the signatures can be cancelled iff a quorum of administrators agree.

One of the new feature, provided by the use of *revocable* blind signature, is that it allows voters to abstain after obtaining a signature at initial registration. Thus, our scheme suits more realistic model of voting.

For n voters, m mix-servers, ℓ administrators, our scheme preserves anonymity in the presence of $(n-2, \frac{m-1}{2}, \frac{\ell-1}{2})^{\mathsf{APA}}$-adversary (see Section 2 for notation). Thus, in this framework, we give more credit to mix-servers than to voters as anonymity is maintained only if either mix-servers do not commit *detectable attacks* that result in disqualification (and invocation of higher layer protocols for punishment). Accordingly, our scheme prevents malicious behavior of mix-servers in a passive way by letting them know that "deviation results in a punishment." If desired, however, our scheme can produce correct output even in the presence of more strong $(n, \frac{m-1}{2}, \frac{\ell-1}{2})^{\mathsf{AAA}}$-adversary (anonymity might be violated, though).

2 Model

There are n voters, m mix-servers, ℓ administrators. We assume that a public-key infrastructure is avail-

* NTT Laboratories Nippon Telegraph and Telephone Corporation 1-1 Hikarinooka, Yokosuka-shi, Kanagawa-ken, 239-0847 Japan

[1] Note that, provided a publicly verifiable robust Mix-net, we can immediately construct a voting scheme without blind signatures.

able so that every entity can establish authentic communication channels. For simplicity, we use a bulletin board on which all transcription of authentic communications appear.

Any party can be a verifier who checks the correctness of the final result. However, in order to convince herself, a verifier must believe that

- all public parameters are correctly chosen so that certain computational assumption holds, and
- at least half of the administrators are honest in such a sence that they will never issue illegitimate signatures, or never leak private signing keys.

To represent the power of adversary, we use the following notation: $(t_v, t_m, t_a)^{\mathtt{APA}}$-adversary means that there are at most t_v actively maricious voters who are under thorough control of the adversary (the first A represents *active* deviation of the voters), at most t_m passively deviating mix-servers whose views are given to the adversary (the second P represents passive deviation of mix-servers), and at most t_a actively maricious administrators (the last A represents active deviation of the administrators). If t_* is not an integer, the nearest smaller integer is taken.

In this paper, we basically consider a poly-time $(n-2, \frac{m-1}{2}, \frac{\ell-1}{2})^{\mathtt{APA}}$-adversary who attempts to violate anonymity or to produce incorrect result. Notice that the bad mix-servers are limited to be passive. However, we also present a measure to recover correct result in the presence of halting or actively deviating mix-servers, privacy may have lost, though.

3 Building Blocks

3.1 Mix-net

We will use a Mix-net that provides the following properties.

- Each server can convince a verifier of the correct relation between a specific pair of messages in its input and output list without endangering the uncertainty of any other combination of messages.
- Anonymous against $(n-2, \frac{m-1}{2}, -)^{\mathtt{AP}-}$-adversary.
- It can carry sufficiently long (e.g. kilobytes of) messages.

The first property will enable *back-trace* of bad results to identify faulty voters or mix-servers.

Classical Mix-nets [7, 17] suffice for the first property. And the second property will be provided by modifying them in a proper way. [2]

The third property would be a problem in practice as formerly designed Mix-nets carry a message within a single block of the underlying public key encryption. Straightforward extension for handling multiple blocks will linearly increase the computational complexity. Better efficiency can be achieved by taking hybrid use of symmetric and asymmetric encryption. We refer to [16] for this type of Mix-net with provable security.

[2] For instance, by forcing each voter to prove his knowledge of the plain message. See [19, 18] for vulnerability of the classical schemes.

3.2 Robust threshold cryptosystem

We assume the use of a secure threshold cryptosystem whose key generation algorithm $\mathcal{G}$ and decryption algorithm $\mathcal{D}$ can be efficiently shared over threshold structure so that decryption can be done if and only if at least a quorum of decryptors are cooperative. Threshold version of the Cramer-Shoup cryptosystem [6, 3] and Threshold Diffie-Hellman cryptosystem of [20] would be the best candidates for our purpose as they provide provable security against adaptive chosen ciphertext attacks and both of them can handle sufficiently long messages.

In the rest of this paper we restrict ourselves staying with higher level description of such cryptosystem by denoting encryption and decryption algorithm by $\mathcal{E}$ and $\mathcal{D}$ respectively.

3.3 Revocable Threshold Blind Signatures

Revocable blind signatures (or fair blind signatures) have been used in secure electronic cash systems in order to prevent perfect crimes [5]. A *threshold* revocable blind signature scheme is a signature scheme with which signatures are issued if and only if a quorum of signers cooperate. Moreover, the blindness can be cancelled if only if a quorum of signers desires so.

The proposed distributed revocable blind signature scheme is derived from the fair blind signature scheme by Camenisch [4]. Let p, q be large primes that satisfy $q|p-1$. Let g be a generator of G_q, which is a subgroup of degree q in Z_p^*. Let g_1 and g_2 be random elements of G_q such that $\log_{g_1} g_2$ is unknown. The signers cooperatively execute the key generation protocol of [11] so that two public keys $y(= g^x)$ and $y_\tau(= g_2^\tau)$ are published and the corresponding private keys x, τ are shared by the signers with $(t+1, n)$-threshold scheme so that signer i privately owns share x_i and τ_i.

Figure 1 illustrates a signature issuing protocol of the proposed scheme. Voter Alice engages in the protocol with at least $\lfloor \frac{\ell-1}{2} \rfloor + 1$ administrators in parallel. Lagrange interpolation coefficient λ_i is defined by the group of administrators Alice accesses.

For the non-interactive proof of the equality of the discrete logarithms, denoted as $NIZK$ in the figure 1, we may use Chaum-Pedersen protocol [8] with Shamir's heuristics [10]. Note that sig_a, which is the output of the protocol, does not solely work as a signature. (Indeed, it does not include msg!) To complete a signature on msg, Alice generates a Schnorr signature sig_u of msg by using a as a secret key and h/g_1 as a public key for base y_τ as illustrated in Figure 2. This second signature is needed not only to involve a message but also to guarantee that h is formed correctly.

If both parties follow the protocol, the resulted sig-

nature (h, y_h, c, s) satisfies verification predicate

$$c \stackrel{?}{=} \mathcal{H}(g\|y\|g^s y^c\|h\|y_h\|h^s {y_h}^c)$$

because

$$\begin{aligned} g^s y^c &= g^{\alpha+\sum \lambda_i \tilde{s}_i} y^{\tilde{c}+\beta} \\ &= g^{\alpha+\sum \lambda_i(\tilde{\omega}_i - \tilde{c}x_i)+\tilde{c}x} y^{\beta} \\ &= g^{\alpha} y^{\beta} g^{\sum \tilde{\omega}_i} \\ &= g^{\alpha} y^{\beta} \prod \tilde{t}_g \\ &= t_g, \end{aligned}$$

holds, and similarly $h^s {y_h}^c = t_h$ holds.

When the verification at the last step of the issuing protocol fails, cheating signers can be identified by publicly checking whether

$$\tilde{t}_{g,i} \stackrel{?}{=} g^{\tilde{s}_i} {y_i}^{\tilde{c}} \text{ and} \quad (1)$$

$$\tilde{t}_{h,i} \stackrel{?}{=} \tilde{h}^{\tilde{s}_i} {y_{h,i}}^{\tilde{c}} \quad (2)$$

holds for every i.

To revoke sig_a, each signer publishes

$$h_i = R^{\tau_i} \quad (3)$$

and proves its correctness by showing $NIZK[\log_d h_i = \log_{g_2} y_{\tau,i}]$. Then h of in the tracing signature tuple can be computed as

$$h = g_1 \prod_i {h_i}^{\lambda_i} \quad (4)$$

because $g_1 \prod_i {h_i}^{\lambda_i} = g_1 \prod_i R^{\lambda_i \tau_i} = g_1 R^{\tau} = g_1 y_{\tau}^{a} = h$. That h will be put in the black-list.

4 Proposed Voting Scheme

Our scheme consists of seven stages: *Preliminary stage*, *Registration stage*, *Vote casting stage*, *Abstention handling stage*, *Mixing stage*, *Screening stage* and *Aggregation stage*. Each stage starts right after the previous stage finishes.

4.1 Preliminary stage

The administrators agree on public parameters p, q, g, g_1 and g_2. Then they generate the following public key pairs in a distributed fashion:

- (y, x) : signing key pair, $y = g^x$
- (y_τ, τ) : revocation key pair, $y_\tau = {g_2}^\tau$
- (K_e, K_d) : encryption and decryption key pair for algorithm $\mathcal{E}$ and $\mathcal{D}$.

Each administrator eventually receives private keys (x_i, τ_i, K_{di}) that corresponds a share of (x, τ, K_d) shared by $(t+1, \ell)$-threshold scheme where $t = \lfloor \frac{\ell-1}{2} \rfloor$. Public parameters $p, q, g, g_1, g_2, y, y_\tau, K_e$ are published authentically.

Each mix-server generates a public and private key pair for underlying encryption scheme. To tolerate halting servers, each server shares its local decryption key among the servers by using $(t'+1, m)$-VSS [9] where $t' = \lfloor \frac{m-1}{2} \rfloor$.

This stage also prepares necessary description of the vote such as the list of candidates, the rule of choice, the voting period, and so on. Let MSP be the message space from which the voters can choose their vote.

4.2 Registration stage

Each voter engages in the blind signature issuing protocol with sufficient number of the administrators as shown in Figure 1. Voter i stores a and sig_a. The administrators store (i, R). If voter declares NG, the administrators takes standard disqualify-then-retry procedure according to the checking equation 1 and 2.

4.3 Vote casting stage

Each voter chooses his vote v from message space MSP and encrypts it with K_e as $enc = \mathcal{E}_{K_e}(v)$. Then, the voter computes sig_u taking enc as a message as shown in Figure 2.

Now, the voter forms correct input message of a Mix-net by taking (sig_a, sig_u, enc) as a plain message so that the tuple will appear at the end of the Mix-net. Let $\mathcal{X}$ be the input message. The user sends $\mathcal{X}$ to the entrance of the Mix-net (a bulletin board) via an authentic channel. Let L_0 be the list of those $\mathcal{X}$-es.

4.4 Abstention handling stage

This stage is performed only if there are voters who have successfully registered but have never casted their vote. For each of such voters, the administrators cooperate to compute tracing information h according to equation 3 and 4 taking R given by the abstained voter at registration. Then if there is an entry in L_m where sig_a contains the h, record the entry (indeed, its position index in L_m) to the tracing list, say $\mathcal{T}$.

4.5 Mixing stage

Taking L_0 from the bulletin board, the mix-servers start processing. The output list, say L_m, is published on the bulletin board.

4.6 Screening stage

This stage consists of the following steps.

1. (Signature verification step) For every entry (sig_a, sig_u, enc) of L_m, the administrators verify the entry by using the checking predicates shown in Figure 2. If an entry fails, record the entry to $\mathcal{T}$.
2. (Back-tracing step) For each entry in $\mathcal{T}$, the following is executed.
   (a) From the last to the first mix-server, each server reveals its input-output relation associated to the tracing entry. If a server fails to show the correct link, the server is disqualified. Otherwise, if all the mix-servers are successful and the trace reaches to a voter, proceed to the next step.

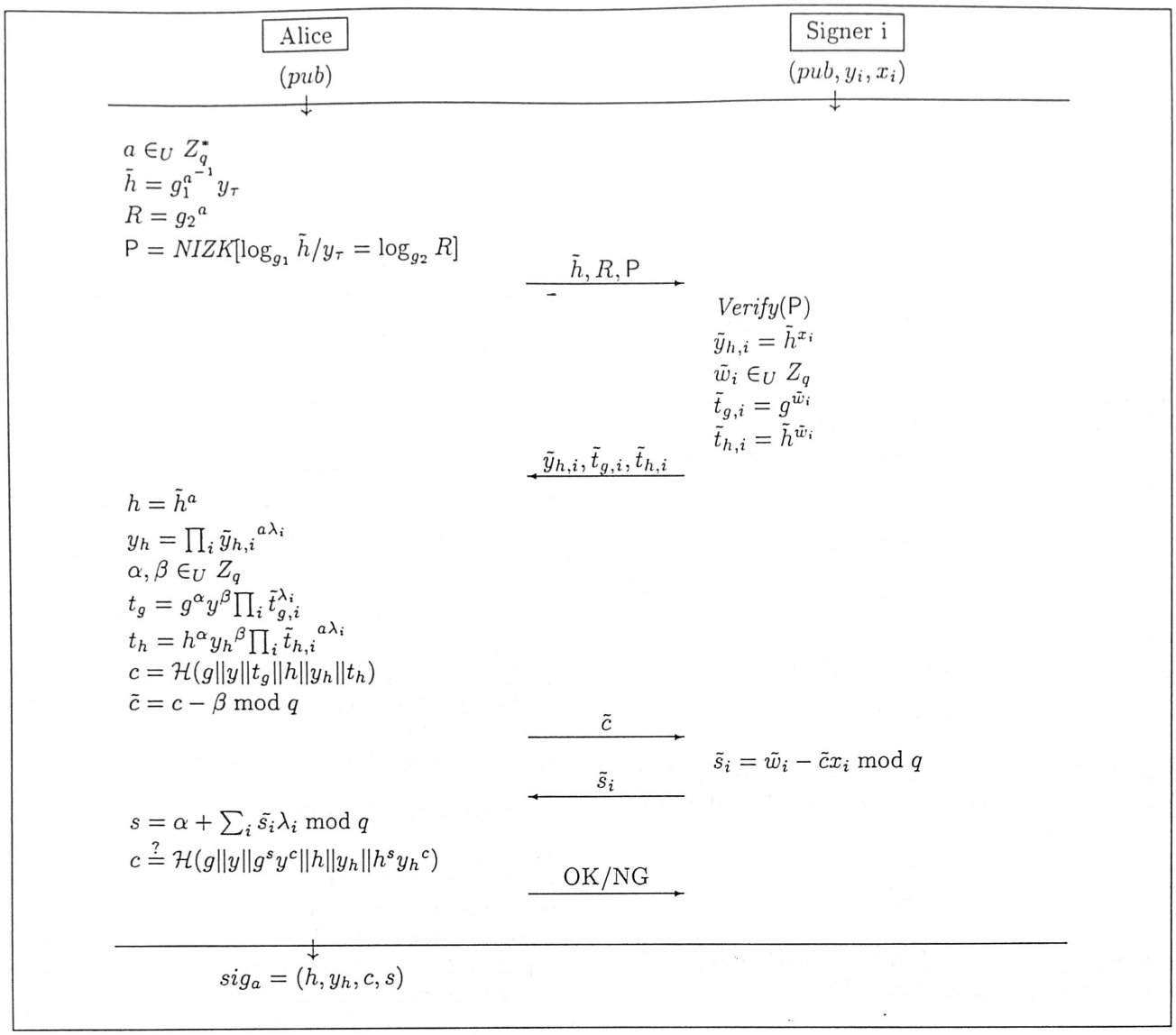

Figure 1: Revocable threshold blind signature issuing protocol. $pub = (p, q, g, g_1, g_2, y, y_\tau)$. Alice executes the protocol with sufficient number of signers in parallel. She stores a privately for later use. The signers store R in case of tracing. A signature on message msg is obtained by doing additional work shown in Figure 2.

(b) The administrators revoke h of the voter. If that h is included in some entry in L_m and it has not yet listed in $\mathcal{T}$, append the entry to $\mathcal{T}$.

The above procedure eventually stops when a server is disqualified or all the entries in $\mathcal{T}$ have been traced. If a server is disqualified, the rest of servers cooperate to disclose its decryption key and perform the decryption in public. Then, the descendent mix-servers reperform their task in the mixing stage. The screening step is replayed from the beginning, too.

4.7 Tallying stage

For each entry of L_m not listed in $\mathcal{T}$, the administrators cooperate to decrypt msg, which actually is $\mathcal{E}(v)$. The resulted list of v is published so that any verifier can see the final result of the vote.

5 Security

Theorem 5.1 *For every correctly formed input $\mathcal{X}$, the corresponding vote v is included in the final list published in the tallying stage in the presence of $(n, \frac{m-1}{2}, \frac{\ell-1}{2})^{\mathtt{AAA}}$-adversary.*

Here, we mean that $\mathcal{X}$ is *correctly formed* if it satisfies $Verify(sig_a, sig_u, enc) = \mathtt{true}$ where $(sig_a, sig_u, enc) = \mathcal{D}_{K_d}(\mathcal{X})$. First suppose that there is a correctly formed entry $\mathcal{X}$ in input list L_0 whose corresponding message (sig_a, sig_u, enc) is not included in L_m. Since at most n valid signatures exists, and the threshold blind signature is assumed to be unforgeable in the presence of at most $\lfloor \frac{\ell-1}{2} \rfloor$ faulty administrators, there is at least one invalid entry in L_m. Then, the screening procedure must detect the actively deviating mix-server. (The trace might once reach to a voter, but eventually it detects faulty mix-server because the revocation will expose a valid signature that differs from the missing entry (sig_a, sig_u, enc). If majority of mix-servers are cooperative, the task of disqualified servers

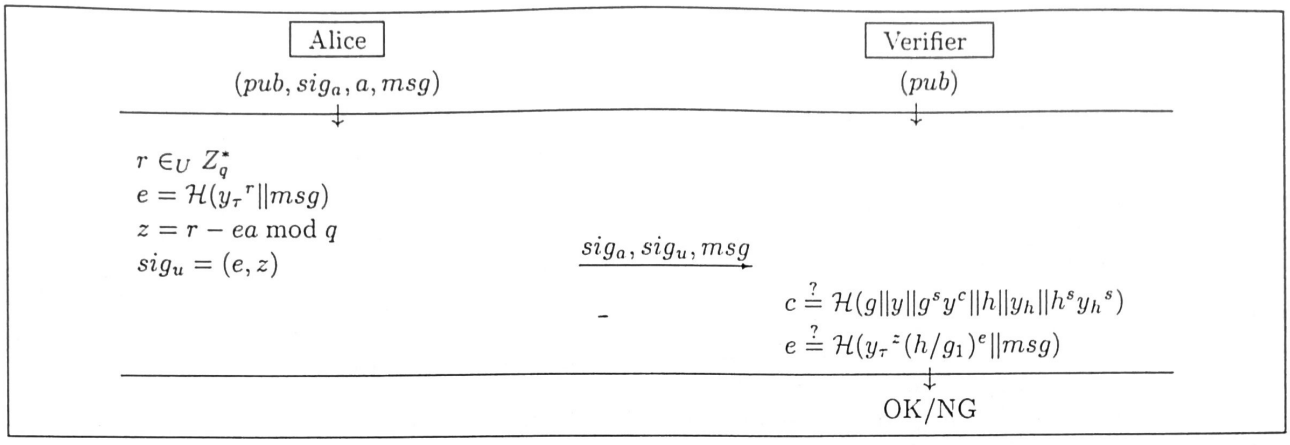

Figure 2: Signature completion and verification. Alice completes the signature by providing a Schnorr signature (e, z) of public key h/g_1 and base y_τ. The verifier is convinced that sig_a is a signature of the signers, and h contains correct information necessary for revocation.

can be done in public as their decryption keys are verifiably shared.) All faulty servers can be eliminated in this way and hence correct (sig_a, sig_u, enc) must appear in L_m. Finally, as we assume robustness of threshold decryption algorithm $\mathcal{D}$, that entry surely yield correct vote $v = \mathcal{D}(enc)$.

Theorem 5.2 *The scheme is fair in such a sense that any poly-time* $(n-1, \frac{m-1}{2}, \frac{\ell-1}{2})^{\mathtt{AAA}}$*-adversary can guess the final result with probability only negligibly better than* $1/|MSP|$ *before the tallying stage starts.*

The above immediately holds assuming that $(\mathcal{E}, \mathcal{D})$ is semantically secure against adaptive chosen ciphertext attack.

Now, the remaining potential fault is that incorrectly formed $\mathcal{X}$ yields a correct tuple (sig_a, sig_u, enc) that satisfies $Verify(sig_a, sig_u, enc) = \mathtt{true}$. This is indeed possible if a voter and some mix-servers collude as they can replace the bad entry with a good one the voter should have cast. However, such a behavior is harmless in practice because the scheme guarantees fairness. That is, the colluding party have to decide whether they turn the bad entry to good or not without seeing other votes. The bottom line is that such an activity is just the same as extending the deadline of the vote casting stage till the end of the mixing stage.

Theorem 5.3 *Against* $(n-2, \frac{m-1}{2}, \frac{\ell-1}{2})^{\mathtt{APA}}$*-adversary, the scheme provides privacy.*

Observe that invalid signature in L_m can happen only if it is sent from a bad voter or it has been replaced in the course of mixing. If a mix replaces a good entry to bad, the mix must fail to show the correct relation in step 2-a of the screening stage. Thus, if no server is disqualified, it is the voter who caused the invalid signature and the back-tracing mechanism always identifies the bad voter. Note that we assume, in Section 3.1, that Mix-servers reveal a specific link without endangering other links. Thus, the screening stage will not threaten the anonymity of honest voters.

6 Conclusion

We have constructed a practical voting scheme by using a simple Mix-net and revocable threshold blind signatures, thus eliminated the use of physical anonymous channels or costly verifiable Mix-nets. The scheme tolerates up to $n-2$ malicious voters, minority of curious mix-servers and minority of malicious administrators. Malicious mix-servers can not violate anonymity without being detected. The scheme also allows voters to abstain after registration as real-world voting does.

References

[1] M. Abe. Universally verifiable mix-net with verification work independent of the number of mix-servers. In K. Nyberg, editor, *Advances in Cryptology — EUROCRYPT '98*, Lecture Notes in Computer Science, pages 437–447. Springer-Verlag, 1998.

[2] M. Abe. Mix-networks on permutation networks. In K. Lam, E. Okamoto, and C. Xing, editors, *Advances in Cryptology - Asiacrypt '99*, volume 1716 of *Lecture Notes in Computer Science*, pages 258–273. Springer-Verlag, 1999.

[3] M. Abe. Robust threshold Cramer-Shoup cryptosystem. In *The Proceedings of the 1999 Symposium on Cryptography and Information Security*, number T1-1.3 in SCIS99, 1999. (in Japanese).

[4] J. Camenisch. *Group Signature Schemes and Payment Systems Based on the Discrete Logarithm Problem.* PhD thesis, ETH Zürich, 1998.

[5] J. Camenisch, J.-M. Piveteau, and M. Stadler. Fair blind signatures. In L. C. Guillou and J.-J. Quisquater, editors, *Advances in Cryptology — EUROCRYPT '95*, volume 921 of *Lecture Notes in Computer Science*, pages 209–219. Springer-Verlag, 1995.

[6] R. Canetti and S. Goldwasser. An efficient threshold public key cryptosystem secure against

adaptive chosen ciphertext attack. In Jacques Stern, editor, *Advances in Cryptology — EUROCRYPT '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 90–106. Springer-Verlag, 1999.

[7] D. L. Chaum. Untraceable electronic mail, return address, and digital pseudonyms. *Communications of the ACM*, 24:84–88, 1981.

[8] D. L. Chaum and T. P. Pedersen. Wallet databases with observers. In E. F. Brickell, editor, *Advances in Cryptology — CRYPTO '92*, volume 740 of *Lecture Notes in Computer Science*, pages 89–105. Springer-Verlag, 1993.

[9] P. Feldman. A practical scheme for non-interactive verifiable secret sharing. In *Proceedings of the 28th IEEE Annual Symposium on Foundations of Computer Science*, pages 427–437, 1987.

[10] A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In A. M. Odlyzko, editor, *Advances in Cryptology — CRYPTO '86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–199. Springer-Verlag, 1986.

[11] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. Secure distributed key generation for discrete-log based cryptosystems. In Jacques Stern, editor, *Advances in Cryptology — EUROCRYPT '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 295–310. Springer-Verlag, 1999.

[12] M. A. Herschberg. Secure electronic voting over the world wide web. Master's thesis, Massachusetts Institute of Technology, 1997.

[13] M. Jakobsson. A practical mix. In K. Nyberg, editor, *Advances in Cryptology — EUROCRYPT '98*, volume 1403 of *Lecture Notes in Computer Science*, pages 448–461. Springer-Verlag, 1998.

[14] M. Jakobsson. Flash mixing. In *PODC99*, 1999.

[15] W. Ogata, K. Kurosawa, K. Sako, and K. Takatani. Fault tolerant anonymous channel. In *ICICS98*, volume 1334 of *Lecture Notes in Computer Science*, pages 440–444. Springer-Verlag, 1998.

[16] M. Ohkubo. Length-preserving hybrid mix. In *The Proceedings of the 2000 Symposium on Cryptography and Information Security*, number B29 in SCIS2K, 2000. (in Japanese).

[17] C. Park, K. Itoh, and K. Kurosawa. Efficient anonymous channel and all/nothing election scheme. In T. Helleseth, editor, *Advances in Cryptology — EUROCRYPT '93*, volume 765 of *Lecture Notes in Computer Science*, pages 248–259. Springer-Verlag, 1994.

[18] B. Pfitzmann. Breaking an efficient anonymous channel. In Alfredo De Santis, editor, *Advances in Cryptology — EUROCRYPT '94*, volume 950 of *Lecture Notes in Computer Science*, pages 339–348. Springer-Verlag, 1995.

[19] B. Pfitzmann and A. Pfitzmann. How to break the direct RSA implementation of MIXes. In J.-J. Quisquater and J. Vandewalle, editors, *Advances in Cryptology - Eurocrypt '89*, volume 434 of *Lecture Notes in Computer Science*, pages 373–381. Springer-Verlag, 1989.

[20] V. Shoup and R. Gennaro. Securing threshold cryptosystems against chosen ciphertext attack. In K. Nyberg, editor, *Advances in Cryptology — EUROCRYPT '98*, Lecture Notes in Computer Science, pages 1–16. Springer-Verlag, 1998.