Vol. 44 No. 8 IPSJ Journal Aug. 2003

Regular Paper

An Unlinkable Divisible Electronic Cash Using Secure Proxy Computation for DL One-way Function

Toru Nakanishi[†] and Yuji Sugiyama[†]

Electronic cash (e-cash) should satisfy the unlinkability and divisibility, for the privacy protection and the convenience, respectively. The unlinkability means the infeasibility of determining whether two payments are made by the same customer. The divisibility means that an e-coin can be divided to spent, and thus the exact payments are available. In the existing unlinkable divisible e-cash system, the customer suffers from the inefficiency of the payment protocol, because of the vast computations of zero-knowledge proofs. This paper proposes an improved system, where the customer's load is vastly reduced. Instead of the customer, the proofs are computed by the active proxies, called trustees. As a side result, the distributed proxy computation protocol for a DL one-way function is also proposed. In the protocol, given a ciphertext of a value a, a quorum of parties verifiably computes a ciphertext of $f(a)$ that is a discrete log type of one-way function, without revealing a and $f(a)$.

1. Introduction

1.1 Background and Previous Works

As the core to realizing the electronic commerce, the electronic cash (e-cash)[1)~9)] is in great demand. In e-cash systems, a customer withdraws electronic *coins* from a bank, and the customer pays the coins to a merchant in the *off-line* manner. The off-line means that the customer has no need to communicate with the bank or a trusted third party during the payment. Finally, the merchant deposits the paid coins to the bank.

To protect the privacy of customers, each payment should be anonymous, and furthermore *unlinkability* should be satisfied. The unlinkability means that any other one except the trusted third party cannot determine whether two payments are made by the same customer. In linkable anonymous e-cash systems, the linked payments enable the other one to trace the payer by other means (i.e., correlating the payments' locality, date, frequency, etc.), as noted by Pfitzmann and Waidner[10)].

In practice, it is desirable that payments of arbitrary amounts can be performed, as well as currently available physical payments. As discussed by Chan, et al.[6)], the known best solution is *divisible* e-cash. The divisibility means that payments of any amount up to the monetary amount of a withdrawn coin can be made. Some divisible e-cash systems have been proposed[2),4),6)], among which, the system[6)] by Chan, et al. is the most efficient. However, these systems do not satisfy the unlinkability among the payments derived from the same coin. Thus, the oftener a customer divides a coin to pay it, the more easily the customer may be traced owing to the linked payments.

As a solution for this problem, Nakanishi and Sugiyama recently proposed an unlinkable divisible electronic cash system[9)], where even the payments derived from the same coin are unlinkable. The key of the solution is the use of the group signature scheme[11)]. Owing to the scheme, during the payment, the customer can prove the ownership of the coin authenticated by the bank without revealing the coin. This means the unforgeability of the paid coin, together with the unlinkability of any pair of payments. On the other hand, to enable the detection of over-spending of the coin, the system furthermore uses a tree, where the root indicates the amount of the withdrawn coin, and any other node indicates a part of the parent's amount. Then, the customer has to pay only the nodes without the ancestor-descendant relationship. For the detection, the customer sends some values, called F values, assigned to the paid nodes during the payment. The F value is computed by applying a sequence of DL (Discrete Log) one-way functions for a secret key correspondent to the coin. To assure the correctness of the computation, the customer proves the sequence of one-way functions in the zero-knowledge fashion. However, for the zero-knowledge proof of the one-way function,

[†] Department of Communication Network Engineering, Faculty of Engineering, Okayama University

the protocol using the so-called cut-and-choose method is only known, and thus the proof protocol is inefficient. Therefore, in this e-cash system, customers with less computational power (e.g., PDA or cellular phone) suffer from the inefficiency of the payment protocol including the proof protocol.

1.2 Contributions

Based on the previous system[9], this paper[☆] proposes an unlinkable divisible e-cash system, where the customer's computations in the payment protocol are vastly reduced, while the computations are conducted by distributed proxy TTPs (trusted third parties) instead. Namely, the idea is that the TTPs compute the F values and prove the correctness. During the payment, the customer sends only the ciphertext of the root F value, together with the proof of the correctness. This proof includes efficient zero-knowledge proofs and a single cut-and-choose zero-knowledge proof. After the merchant sends the bank the ciphertext for the deposit, the bank requests the TTPs the proxy computations to obtain the F values. This computations are conducted as the secure multiparty computation via ElGamal ciphertexts[13]. This means that any quorum of the TTPs can compute the correct result, while the TTPs not containing the quorum can not obtain any information. Therefore, the introduce of the proxy computation does not influence the security. On the other hand, the payment protocol is more efficient, since the inefficient proof is used once during a payment transaction (The previous payment protocol uses inefficient proofs about T times, where T is the product of the number of all the paid nodes and the depth of the nodes). The above secure multiparty computation requires the secure multiparty computation for a DL one-way function. However, the protocol to accomplish this is unknown. Therefore, this paper also proposes the concrete secure multiparty computation protocol for the DL one-way function.

The organization of this paper is as follows: In Section 2, the requirements on e-cash systems are reviewed. In Section 3, the used known cryptographic tools are reviewed. In Section 4, a secure proxy computation protocol for the DL one-way function is constructed. In Section 5, an improved unlinkable divisible e-cash systems is proposed, as followed by the security and efficiency discussions. Finally, we conclude this paper in Section 6.

2. Model and Requirements

As well as the previous system, we adopt the model of the escrow e-cash[5],[7], where trusted third parties, called trustees, participate in the system. The trustees can cooperatively revoke the anonymity of the payments to protect illegal acts of anonymous customers, such as money laundering, blackmailing attack[14] and so on. Thus, the escrow e-cash model equips a tracing protocol to revoke the anonymity, in addition to setup, withdrawal, payment, deposit protocols equipped in the conventional e-cash model. When the revocation is requested by a court for example, due to an illegal act, the trustees cooperatively execute this tracing protocol to revoke the anonymity. To make the trust high, this model adopts multiple trustees, which are trusted in the threshold setting. Namely, it is trusted that any quorum (more than a designated threshold) of trustees is not corrupted. The trust is concerned with only the protection of the customers' privacy. That is, if the quorum colludes, the privacy (the anonymity and unlinkability below) is compromised. Of course, the merchants tend to know the private information, and in addition the banks may want the information since they are also commercial presence. Therefore, though the bank may be a member of the trustees, noncommercial third parties should be included in the trustees, such as ombudsmans. The previous system[9] is also on this model, though only the case of a single trustee is concretely described for simplicity.

Our escrow e-cash model has a difference: The previous system uses the passive trustees, that is, they act only in case of dispute. However, to accomplish customer's less computations, this paper uses the active trustees, that is, they always act in the e-cash protocols. Concretely, in all the deposit protocols, they instead perform the computations that customers should perform in the previous system. The trust for the active trustees is the same as the previous model. Namely, the trust is concerned with only the customers' privacy in the threshold setting. Therefore, as for only the trust, the reality to set up the trustees is the same as the previous model. On the other hand, the proxy computations may bring the problem on the scalability. The problem and the reality

[☆] A preliminary version of the paper appeared in ISITA2002[12].

concerned with the scalability are discussed in Section 5.4.

In our model, the trustees are assigned to two types of tasks, as mentioned above: The anonymity revocations and the proxy computations. If only the trust is considered, it is not too meaning to partition the trustees into the passive trustees to perform the revocation and the active trustees to perform the proxy computations. However, these two types of tasks require the different managements and the proxy computations may have the problem on the scalability. Thus, in such a sense, the partition is sufficiently meaning. However, for simplicity, we concentrate on the model where a single group of active trustees with the threshold trust manage the both tasks.

The requirements for divisible e-cash systems are as follows[9]:

Unforgeability: A coin and a transcript of a payment can not be forged.

No over-spending: The customer who overspends a coin can be identified.

No swindling: No one except the customer who withdraws a coin can spend the coin. The deposit information can not be forged.

Anonymity: No one except the payer and the quorum of trustees can trace the payer from the payment.

Unlinkability: No one except the payer and the quorum of trustees can determine whether any pair of payments is executed by the same customer, unless the payments cause over-spending.

Anonymity revocation: Anonymity of a transcript of a payment can be revoked only by the quorum of trustees and when necessary, where the following revocation procedures should be accomplished:

- **Owner tracing:** To identify the payer of a targeted payment.
- **Coin tracing:** To link a targeted withdrawal of a coin to the payments derived from the coin.

Divisibility: Payments of any amount up to the monetary amount of a withdrawn coin can be made.

Off-line-ness: During payments, the payer communicates only with the merchant.

3. Building Blocks

3.1 Signature of Knowledge

As building blocks, the previous system uses signatures converted by so-called Fiat-Shamir heuristic[15] from honest-verifier zero-knowledge proofs of knowledge, which is called as signatures of knowledge. We abbreviate them as SPKs. The SPKs are secure in the random oracle model[16], if the underlying interactive protocols are the zero-knowledge proofs of knowledge. The SPKs are denoted as

$$\mathrm{SPK}\{(\alpha,\beta,\ldots):R(\alpha,\beta,\ldots)\}(m),$$

which means the signature for message m by a signer with the secret knowledge $\alpha,\beta,\ldots$ satisfying the relation $R(\alpha,\beta,\ldots)$. In this notation, the Greek letters denote the signer's secret knowledge, and other parameters denote public values. Let $\mathcal{G}=\langle g\rangle$ be a cyclic group of order p, which is a subgroup of $Z_{p'}^*$ for a prime p' satisfying $p|(p'-1)$. Let $\mathcal{G}'=\langle g'\rangle$ be a cyclic group of order p', which is a subgroup of $Z_{p''}^*$ for a prime p'' satisfying $p'|(p''-1)$. Then, the following SPKs are used. For the concrete constructions, refer to the previous paper[9].

SPK of representations: An SPK of representations of $y_1,\ldots,y_w\in\mathcal{G}$ to bases $g_1,\ldots,g_v\in\mathcal{G}$ is denoted as

$$\mathrm{SPK}\ \{(\alpha_1,\ldots,\alpha_u):(y_1=\prod_{j=1}^{\ell_1}g_{b_{1j}}^{\alpha_{e_{1j}}})$$
$$\wedge\cdots\wedge(y_w=\prod_{j=1}^{\ell_w}g_{b_{wj}}^{\alpha_{e_{wj}}})\}(m),$$

where constants $\ell_i\in\{1,\ldots v\}$ indicate the number of bases of y_i, the indices $e_{ij}\in\{1,\ldots,u\}$ refer to the elements $\alpha_1,\ldots,\alpha_u$ and the indices $b_{ij}\in\{1,\ldots,v\}$ refer to the bases $g_1,\ldots,g_v$.

SPKs of e-th root of representation: An SPK of the e-th root of the DL of $y\in\mathcal{G}$ to the base $g\in\mathcal{G}$ on m is denoted as

$$\mathrm{SPK}\{\beta:y=g^{\beta^e}\}(m).$$

An *SPK* of the e-th root of the g-part of a representation of $y\in\mathcal{G}$ to the bases $h,g\in\mathcal{G}$ on m is denoted as

$$\mathrm{SPK}\{(\gamma,\delta):y=h^\gamma g^{\delta^e}\}(m).$$

SPK of the same DL as a double DL: The double discrete logarithm (double DL) of $y'\in\mathcal{G}'$ to the bases $g'\in\mathcal{G}'$ and $h\in\mathcal{G}$ implies $x\in Z_p$ satisfying $y'=g'^{(h^x)}$ if such an x exists. Then, an SPK of the DL of $y\in\mathcal{G}$ to the base $g\in\mathcal{G}$ and the double DL of y' to the bases g' and h on m, where the DL equals the double DL, is denoted as

$$\mathrm{SPK}\{\epsilon : y = g^{\epsilon} \wedge y' = g'^{(h^{\epsilon})}\}(m).$$

Only the last SPK needs vast exponentiations, about 40 exponentiations, because of using so-called cut-and-choose method, as shown in the paper[9]. The previous payment protocol suffers from the inefficiency of this SPK.

3.2 Threshold ElGamal Cryptosystem

In addition, we use the threshold ElGamal cryptosystem[17] as the trunk of the improvement. Thus, here we review the cryptosystem in detail. The ElGamal ciphertext for message $m \in \mathcal{G}$ with the secret key $x \in_R Z_p$ and the public key $y = g^x$ is computed as $(G = g^r, Y = my^r)$ where $r \in_R Z_p$. In the threshold setting, the secret key x is shared among the multiparty, that is the trustees in this application. Assume that the trustee T_i is numbered as $i = 1, \ldots, N$. Each T_i keeps his share x_i of x secret, and publishes his public key $y_i = g^{x_i}$. We use Shamir secret sharing scheme, where x equals $\Sigma_{i \in Q} x_i L_{(Q,i)}$ (mod p), for any $Q \subseteq \{1, \ldots, N\}$ satisfying $|Q| \geq K$ with the threshold K, and each Lagrange coefficient $L_{(Q,i)} = \prod_{j \in Q, j \neq i} \frac{j}{j-i}$. As for the protocol to distribute the secret, refer to Pedersen's[18] and Gennaro *et al.*'s[19] papers. In this setting, we use the following scheme used in many literatures (e.g., Abe[20], Jakobsson and Juels[13]):

Verifiable threshold decryption: Trustees can cooperatively decrypt the ciphertext (G, Y) as follows: T_i publishes $G_i = G^{x_i}$. Furthermore, T_i proves the correctness by publishing $\mathrm{SPK}\{\alpha : G_i = G^{\alpha} \wedge y_i = g^{\alpha}\}(\tilde{0})$, where $\tilde{0}$ is the empty string. When the SPK is not accepted, the corresponding T_i is removed as dishonest. Let Q be the set of indices of T_i not removed. Then, anyone can decrypt the ciphertext by computing $Y / \prod_{i \in Q} G_i^{L_{(Q,i)}} = Y / \prod_{i \in Q} G^{L_{(Q,i)} x_i} = Y / G^{\sum_{i \in Q} L_{(Q,i)} x_i} = m$.

4. Secure Proxy Computation for DL One-way Function

In the construction of the proposed e-cash system, we use a one-way function based on the DL, that is, it is $f(a) = h^a$ for an element h of a cyclic group $\mathcal{G}$, and $a \in Z_{|\mathcal{G}|}$. Furthermore, we need for the distributed proxy servers to cooperatively compute $Enc(f(a))$ from $Enc(a)$ without revealing a and $f(a)$, where Enc is an ElGamal encryption. This section provides the protocol, together with the security consideration.

4.1 Definitions

The proxy computation for the one-way function via ciphertexts is abbreviated as PCOWF. Let $\mathcal{G}$ be a cyclic group of order p, which is a subgroup of $Z_{p'}^*$ for a prime p' satisfying $p|(p'-1)$. Let $\mathcal{G}'$ be a cyclic group of order p', which is a subgroup of $Z_{p''}^*$ for a prime p'' satisfying $p'|(p''-1)$.

Definition 1 The participants of PCOWF protocol is N proxy servers $T_1, \ldots, T_N$ that share secret keys of the ElGamal threshold cryptosystems on the groups $\mathcal{G}$ and $\mathcal{G}'$. In the PCOWF protocol, the server's common public input is $(g, y, y_1, \ldots, y_N, g', y', y'_1, \ldots, y'_N, h', (G, Y))$, where g (resp., g') is a generator of $\mathcal{G}$ (resp., $\mathcal{G}'$), $y, y_1, \ldots, y_N$ (resp., $(y', y'_1, \ldots, y'_N)$) are the whole public key and $T_1, \ldots, T_N$'s individual public keys of the threshold cryptosystem on the group $\mathcal{G}$ (resp., $\mathcal{G}'$), h' is an element of $\mathcal{G}'$, and (G, Y) is an ElGamal ciphertext of unknown $a \in \mathcal{G}$, w.r.t. y. The private input of T_i is (x_i, x'_i), where x_i (resp., x'_i) is his share of the secret key of the threshold ElGamal cryptosystem on $\mathcal{G}$ (resp., $\mathcal{G}'$). The common output is an ElGamal ciphertext (G', Y') of h'^a w.r.t. y'.

Definition 2 *Secure* PCOWF protocol satisfies the following properties.

Completeness: Any quorum of honest proxy servers can complete the protocol.

Robustness: Any server who disobeys the protocol is detected by the honest servers. This property allows the honest servers to remove the dishonest server for completing the correct computation.

Public verifiability: Anyone can verify that the output is computed correctly.

Privacy: The protocol view does not leak any information about a and h^a, even if any set of servers smaller than the quorum colludes.

4.2 SPK of DL of Representation

We define the following SPK used in the proposed PCOWF protocol.

Definition 3 An SPK of a representation of $z' \in \mathcal{G}'$ to the bases $g', h' \in \mathcal{G}'$, and furthermore of the DL of the h' part of the representation to the base $h \in \mathcal{G}$, which equals the DL of $z \in \mathcal{G}$ to the base $g \in \mathcal{G}$, on message m, is denoted as

$$\mathrm{SPK}\{(\alpha,\beta): z' = g'^{\alpha} h'^{(h^{\beta})} \wedge z = g^{\beta}\}(m).$$

Now, we describe only the interactive version of this SPK protocol, since Fiat-Shamir heuristic[15] converts the interactive one into the corresponding SPK. Let k be a security parameter for the forgery probability 2^{-k}.

Interactive version of the SPK:

Repeat k times the following steps:

(1) The prover chooses $r_1 \in_R Z_{p'}$ and $r_2 \in_R Z_p$. Then, the prover sends $t_1 = g'^{r_1} h'^{h^{r_2}}$ and $t_2 = g^{r_2}$ to the verifier.

(2) The verifier returns $c \in_R \{0,1\}$.

(3) If $c = 0$, the prover sends $s_1 = r_1$ and $s_2 = r_2$. The verifier checks $t_1 = g'^{s_1} h'^{h^{s_2}}$ and $t_2 = g^{s_2}$.
Otherwise, the prover sends $s_1 = r_1 - \alpha h^{r_2-\beta} \pmod{p'}$, and $s_2 = r_2 - \beta \pmod{p}$. The verifier checks $t_1 = g'^{s_1} z'^{h^{s_2}}$ and $t_2 = g^{s_2} z$. □

Theorem 1 The above protocol is an any-verifier zero-knowledge proof of knowledge of (α,β) such that $z' = g'^{\alpha} h'^{(h^{\beta})}$ and $z = g^{\beta}$.

Proof:

The completeness in the case of $c = 1$ holds because of the equations

$$\begin{aligned} g'^{s_1} z'^{h^{s_2}} &= g'^{r_1 - \alpha h^{r_2-\beta}} (g'^{\alpha} h'^{h^{\beta}})^{h^{r_2-\beta}} \\ &= g'^{r_1 - \alpha h^{r_2-\beta}} g'^{\alpha h^{r_2-\beta}} h'^{h^{\beta} h^{r_2-\beta}} \\ &= g'^{r_1} h'^{h^{r_2}} = t_1, \end{aligned}$$

and

$$g^{s_2} z = g^{r_2-\beta} g^{\beta} = g^{r_2} = t_2.$$

The case of $c = 0$ is straightforward.

For the soundness, it is sufficient to extract secrets from the two accepting triples $(t_1, t_2, c = 0, s_1, s_2)$ and $(t_1, t_2, \tilde{c} = 1, \tilde{s}_1, \tilde{s}_2)$, as follows: Assume $z' = g'^{\alpha} h'^{h^{\beta}}$ and $z = g^{\beta}$. Then, $t_1 = g'^{s_1} h'^{h^{s_2}}$ and $t_1 = g'^{\tilde{s}_1} z'^{h^{\tilde{s}_2}} = g'^{\tilde{s}_1} (g'^{\alpha} h'^{h^{\beta}})^{h^{\tilde{s}_2}} = g'^{\tilde{s}_1 + \alpha h^{\tilde{s}_2}} h'^{h^{\beta+\tilde{s}_2}}$. On the other hand, $t_2 = g^{s_2}$ and $t_2 = g^{\tilde{s}_2} z = g^{\tilde{s}_2} g^{\beta} = g^{\tilde{s}_2+\beta}$. Thus, $s_1 = \tilde{s}_1 + \alpha h^{\tilde{s}_2} \pmod{p'}$ and $s_2 = \beta + \tilde{s}_2 \pmod{p}$ hold. Thus, we can extract α by $(s_1 - \tilde{s}_1)/h^{\tilde{s}_2} \pmod{p'}$, and β by $s_2 - \tilde{s}_2 \pmod{p}$.

Finally, we show the zero-knowledgeness. Consider the following simulator. For any round, the simulator guesses $c' \in_R \{0,1\}$. If $c' = 0$, the simulator obeys the protocol honestly, whose view is statistically indistinguishable from the real one. If $c' = 1$, the simulator chooses $\dot{s}_1 \in_R Z_{p'}$ and $\dot{s}_2 \in_R Z_p$, and sends $\dot{t}_1 = g'^{\dot{s}_1} z'^{h^{\dot{s}_2}}$ and $\dot{t}_2 = g^{\dot{s}_2} z$. When receiving $c = 1$ (If $c = 0$, rewind the simulator), the simulator sends $\dot{s}_1$ and $\dot{s}_2$, which satisfies the verification equations. Then, note that $\dot{t}_1 = g'^{\dot{s}_1 + \alpha h^{\dot{s}_2}} h'^{h^{\beta+\dot{s}_2}}$ and $\dot{t}_2 = g^{\beta+\dot{s}_2}$ hold. When $\dot{r}_1 = \dot{s}_1 + \alpha h^{\dot{s}_2} \pmod{p'}$ and $\dot{r}_2 = \beta + \dot{s}_2 \pmod{p}$ are set, $\dot{r}_1$ and $\dot{r}_2$ independently distribute uniformly over $Z_{p'}$ and Z_p, respectively. Thus, the view of the simulator $(\dot{t}_1 = g'^{\dot{r}_1} h'^{h^{\dot{r}_2}}, \dot{t}_2 = g^{\dot{r}_2}, c = 1, \dot{s}_1 = \dot{r}_1 - \alpha h^{\dot{r}_2-\beta} \pmod{p'}, \dot{s}_2 = \dot{r}_2 - \beta \pmod{p})$ is statistically indistinguishable from the real protocol's view. □

4.3 Proposed Protocol

We describe the PCOWF protocol in detail.

PCOWF protocol:

For simplicity, assume that the quorum of servers consists of $T_1, \ldots, T_K$. For the quorum, let the Lagrange coefficient of each T_i be L_i. The protocol is sequentially executed by servers $T_1, \ldots, T_K$. Note that the input ciphertext of a is (G, Y).

(1) T_1 first chooses $r_1 \in Z_{p'}$, and publishes $(G'_1, Y'_1) = (g'^{r_1}, y'^{r_1} (h'^{Y})^{(G^{-L_1})^{x_1}})$. Furthermore, T_1 publishes $\mathrm{SPK}\{(\alpha_1, \beta_1) : G'_1 = g'^{\alpha_1} \wedge Y'_1 = y'^{\alpha_1} (h'^{Y})^{\beta_1}\}(\tilde{0})$ and $\mathrm{SPK}\{(\gamma_1, \delta_1) : Y'_1 = y'^{\gamma_1} (h'^{Y})^{(G^{-L_1})^{\delta_1}} \wedge y_1 = g^{\delta_1}\}(\tilde{0})$, where $\tilde{0}$ denotes the empty message.

(2) For $2 \le i \le K$, let the output of T_{i-1} be (G'_{i-1}, Y'_{i-1}). Any other T_i $(2 \le i \le K)$ chooses $r_i \in Z_{p'}$, and publishes $(G'_i, Y'_i) = (g'^{r_i} G'^{(G^{-L_i})^{x_i}}_{i-1}, y'^{r_i} Y'^{(G^{-L_i})^{x_i}}_{i-1})$. Furthermore, T_i publishes $\mathrm{SPK}\{(\alpha_i, \beta_i) : G'_i = g'^{\alpha_i} G'^{\beta_i}_{i-1} \wedge Y'_i = y'^{\alpha_i} Y'^{\beta_i}_{i-1}\}(\tilde{0})$ and $\mathrm{SPK}\{(\gamma_i, \delta_i) : G'_i = g'^{\gamma_i} G'^{(G^{-L_i})^{\delta_i}}_{i-1} \wedge y_i = g^{\delta_i}\}(\tilde{0})$.

The output (G', Y') of the protocol is T_K's output (G'_K, Y'_K). □

Remark 1 This protocol must be sequentially executed by trustees, though the original threshold decryption[13),20)] or threshold DSS signature generation[21)] etc. does not need such a sequential execution. This limitation makes the removal of the dishonest trustee more delicate. If the SPK published by a trustee is not accepted, the other honest trustees should stop the current protocol. Then, Q is changed such that the dishonest trustee is removed, and they should restart from the beginning of this protocol.

4.4 Security

We discuss the security of the proposed protocol.

Completeness: Let x' be the secret key correspondent to y'. Note that $y' = g'^{x'}$. The T_1's output (G'_1, Y'_1) satisfies the equations

$$(Y'_1/G'^{x'}_1) = (y'^{r_1}(h'^{Y})^{(G^{-L_1})^{x_1}})/(g'^{r_1})^{x'}$$
$$= (y'^{r_1}h'^{Y/G^{L_1 x_1}})/y'^{r_1} = h'^{Y/G^{L_1 x_1}}.$$

Then, the next T_2's output (G'_2, Y'_2) satisfies the equations

$$\begin{aligned} Y'_2/G'^{x'}_2 &= \frac{y'^{r_2}Y'^{(G^{-L_2})^{x_2}}_1}{(g'^{r_2})^{x'}(G'^{x'}_1)^{(G^{-L_2})^{x_2}}} \\ &= \frac{y'^{r_2}Y'^{(G^{-L_2})^{x_2}}_1}{y'^{r_2}(G'^{x'}_1)^{(G^{-L_2})^{x_2}}} \\ &= (Y'_1/G'^{x'}_1)^{(G^{-L_2})^{x_2}} \\ &= (h'^{Y/G^{L_1 x_1}})^{(G^{-L_2})^{x_2}} \\ &= h'^{Y/G^{L_1 x_1 + L_2 x_2}}. \end{aligned}$$

The similar equations hold for the other T_i's output. Therefore, the last output (G', Y') satisfies the equations $Y'/G'^{x'} = Y'_K/G'^{x'}_K = h'^{Y/G^{L_1 x_1 + \cdots + L_K x_K}} = h'^{a}$. This implies that (G', Y') is a ciphertext of h'^{a}.

Robustness and public verifiability: To satisfy these properties, T_i only has to prove the knowledge of (α, β) such that $G'_1 = g'^{\alpha} \wedge Y'_1 = y'^{\alpha}(h'^{Y})^{(G^{-L_1})^{\beta}} \wedge y_1 = g^{\beta}$ if $i = 1$, or $G'_i = g'^{\alpha}G'^{(G^{-L_i})^{\beta}}_{i-1} \wedge Y'_i = y'^{\alpha}Y'^{(G^{-L_i})^{\beta}}_{i-1} \wedge y_i = g^{\beta}$ otherwise. Consider only the case of $i \geq 2$, since the same discussion holds for $i = 1$. In the PCOWF protocol, T_i performs $\mathrm{SPK}\{(\alpha_i, \beta_i) : G'_i = g'^{\alpha_i}G'^{\beta_i}_{i-1} \wedge Y'_i = y'^{\alpha_i}Y'^{\beta_i}_{i-1}\}(\tilde{0})$ and $\mathrm{SPK}\{(\gamma_i, \delta_i) : G'_i = g'^{\gamma_i}G'^{(G^{-L_i})^{\delta_i}}_{i-1} \wedge y_i = g^{\delta_i}\}(\tilde{0})$. Thus, from two accepting triples of these SPKs, we can extract $(\alpha_i, \beta_i, \gamma_i, \delta_i)$ satisfying the above predicates. Then, for the extracted values, $G'_i = g'^{\alpha_i}G'^{\beta_i}_{i-1} = g'^{\gamma_i}G'^{(G^{-L_i})^{\delta_i}}_{i-1}$, and thus $\alpha_i = \gamma_i$ and $\beta_i = (G^{-L_i})^{\delta_i}$, since it is infeasible to compute two representations to the same bases. Therefore, the pair (γ_i, δ_i) satisfies $G'_i = g'^{\gamma_i}G'^{(G^{-L_i})^{\delta_i}}_{i-1}$, $Y'_i = y'^{\gamma_i}Y'^{(G^{-L_i})^{\delta_i}}_{i-1}$, and $y_i = g^{\delta_i}$ This implies that these SPKs prove the knowledge of the wanted (α, β).

Privacy: Since the used SPKs do not reveal any information, consider only (G'_i, Y'_i). As inspired by the discussion for the completeness, each (G'_i, Y'_i) is a random ciphertext of $h'^{Y/G^{\Sigma_{1 \leq j \leq i} L_j x_j}}$ w.r.t. public key y'. Furthermore, all $T_1, \ldots, T_i$ provide the randomness of the ciphertext. Thus, the colluding servers not containing a quorum obtain only $h'^{Y/G^{\Sigma_{1 \leq j \leq \tilde{i}} L_j x_j}}$ for $\tilde{i} < K$ or values computed from a ciphertext of an unknown value. Therefore, the colluding servers obtain no useful information on a and $f(a)$.

5. Improved Unlinkable Divisible E-cash System

5.1 Basic Idea

Our system is based on the previous system[9], which uses the group signature scheme[11]. The group signature scheme allows a group member to anonymously sign on a group's behalf. Furthermore, the anonymity of the signature can be revoked by the trusted party. In the scheme[11], the group consists of owners of unforgeable certificates issued from the group manager. In the previous e-cash system, the certificate is used as a coin issued from the bank and the group signature is used as a payment transcript. This simple replacement brings the system the anonymity, unlinkability, unforgeability, no swindling, off-line-ness, and owner tracing of the anonymity revocation. Furthermore, in the previous system, mechanisms to enable coin tracing and to detect over-spending of a coin are added. The former mechanism is that, in a withdrawal, a customer is forced to send the ciphertext of a value, which is linked to payments derived from the withdrawal, with the trustees' key.

To protect over-spending, the previous system uses the tree approach. For simplicity, we describe only the system using a binary tree, though we can also construct the system using a general tree with three or more children, such as the previous system. The withdrawn coin is assigned to a tree, where the root indicates the monetary amount of the coin, and any other node indicates the half amount of the parent. In this situation, to protect over-spending, a customer can spend only nodes without the ancestor-descendant relationship. For the detection, the customer has to send some values, called F values, assigned to the nodes with the paid amount during the payment. The F value of the root node is proper to the coin, which is concretely a value applied by a DL one-way function for a secret key in the group signature. The F value of any other node is applied by a DL one-way function for F value of the parent node. Thus, the nodes which have the ancestor-descendant relationship can be linked by a sequence of functions, while the nodes without the relationship do not have such a sequence. However, to assure the correctness of the F value, the customer has to prove the sequence of one-way functions in the zero-knowledge fashion. However, for the zero-knowledge proof of the one-way function, the protocol using the so-called cut-and-choose method is only known, and thus the proof protocol is inefficient. Therefore, customers with less computational power suffer from the inefficiency of the payment protocol.

In the proposed system, instead of sending the F values and the zero-knowledge proofs, the customer sends only the verifiable ElGamal ciphertext of the root F value. Therefore, the payment protocol is much more efficient, though the mechanism of coin tracing needs a single cut-and-choose zero-knowledge proof, as well as the previous protocol. After the merchant sends the bank the payment transcript for the deposit, the bank requests the trustees to compute the F values and the proofs from the ciphertext in the payment transcript, as the proxy. The trustees cooperatively compute them with the public verifiability and the privacy, using the PCOWF protocol multiple times and using the verifiable threshold decryption. From the result, the bank can check overspending, as well as the previous system.

5.2 Proposed System

5.2.1 Setup

This is the similar to the previous. The difference is the setup of the threshold ElGamal cryptosystem. Notations are depicted in **Table 1**.

Table 1 Notations in proposed e-cash protocols.

w	a monetary amount of a coin
ℓ	a depth in a tree for the divisibility with $w = 2^{\ell-1}$
N	number of trustees
n	RSA modulus
e_1, e_2, f_1, f_2	public parameters for e-coins
p_2	a prime with $n \vert (p_2 - 1)$
p_i $(3 \leq i \leq \ell + 1)$	a prime with $p_{i-1} \vert (p_i - 1)$
$\mathcal{G}_n$, g_n	a subgroup with order n of $Z^*_{p_2}$ and its generator
$\mathcal{G}_{p_i}$, g_{p_i} $(2 \leq i \leq \ell)$	a subgroup with order p_i of $Z^*_{p_{i+1}}$ and its generator
$h, \tilde{h}$	public bases from $\mathcal{G}_n$
$h_{(i,0)}, h_{(i,1)}$ $(2 \leq i \leq \ell)$	public bases from $\mathcal{G}_{p_i}$
$y_n, y_{(n,1)}, \cdots, y_{(n,N)}$	ElGamal public keys of trustees on $\mathcal{G}_n$
$y_{p_i}, y_{(p_i,1)}, \cdots, y_{(p_i,N)}$ $(2 \leq i \leq \ell)$	ElGamal public keys of trustees on $\mathcal{G}_{p_i}$

(1) The bank decides the monetary amount of this coin, $w = 2^{\ell-1}$, for a positive integer ℓ. Then, the bank computes an RSA modulus n, two public exponents $e_1, e_2 > 1$, and two integers $f_1, f_2 > 1$. The choices are discussed by Camenisch and Stadler [11]. Then, the bank chooses a cyclic group $\mathcal{G}_n = \langle g_n \rangle$ of order n which is a subgroup of $Z^*_{p_2}$ for a prime p_2 satisfying $n|(p_2-1)$. Similarly, the bank chooses a cyclic group $\mathcal{G}_{p_i} = \langle g_{p_i} \rangle$ of order p_i which is a subgroup of $Z^*_{p_{i+1}}$ for a prime p_{i+1} satisfying $p_i|(p_{i+1}-1)$ with all i $(2 \leq i \leq \ell)$. Furthermore, the bank chooses elements $h, \tilde{h} \in \mathcal{G}_n$, $h_{(2,0)}, h_{(2,1)} \in \mathcal{G}_{p_2}, \ldots, h_{(\ell,0)}, h_{(\ell,1)} \in \mathcal{G}_{p_\ell}$ whose DL to the bases $g_n, g_{p_2}, \ldots, g_{p_\ell}$ are unknown, respectively. Finally, the bank publishes $\mathcal{Y} = (n, e_1, e_2, f_1, f_2, \mathcal{G}_n, \mathcal{G}_{p_2}, \ldots, \mathcal{G}_{p_\ell}, g_n, g_{p_2}, \ldots, g_{p_\ell}, h, \tilde{h}, h_{(2,0)}, \ldots, h_{(\ell,0)}, h_{(2,1)}, \ldots, h_{(\ell,1)})$ as the public key, and keeps the factorization of n secret.

(2) N trustees $T_1, \ldots, T_N$ cooperatively set up keys of the threshold ElGamal cryptosystems on the groups $\mathcal{G}_n, \mathcal{G}_{p_2}, \ldots, \mathcal{G}_{p_\ell}$. Let the public key $(y, y_1, \ldots, y_N)$ of the cryptosystem on the group $\mathcal{G}_n$ (resp., $\mathcal{G}_{p_i}$) be $(y_n, y_{(n,1)}, \ldots, y_{(n,N)})$ (resp., $(y_{p_i}, y_{(p_i,1)}, \ldots, y_{(p_i,N)})$). Then, the DL of y_n to the base g_n is the secret key, which is shared by each trustee T_i as the DL of $y_{(n,i)}$ to the base g_n. The cases of all $\mathcal{G}_{p_i}$ are similar.

5.2.2 Withdrawal

This is the same as the previous.

(1) A customer chooses $x_C \in_R Z^*_n$ to compute $y_C = x_C^{e_1} \bmod n$ and $z_C = h^{y_C}$. Then, the customer chooses $r_1, r_2 \in_R Z^*_n$ to compute $\tilde{y}_C = r_1^{e_2}(f_1 y_C + f_2) \bmod n, \dot{G}_n = g_n^{r_2}$, and $\dot{Y}_n = y_n^{r_2} \tilde{h}^{y_C}$. Furthermore, the customer computes the following SPKs:

$$V_1 = \mathrm{SPK}\{\alpha : z_C = h^{\alpha^{e_1}}\}(\tilde{0}),$$

$$V_2 = \mathrm{SPK}\{\beta : h^{\tilde{y}_C} = (z_C^{f_1} h^{f_2})^{\beta^{e_2}}\}(\tilde{0}),$$
$$V_3 = \mathrm{SPK}\{(\gamma, \delta) : \dot{G}_n = g_n^{\gamma} \wedge \dot{Y}_n = y_n^{\gamma} \tilde{h}^{\delta} \wedge z_C = h^{\delta}\}(\tilde{0}).$$

The customer sends the bank $(\tilde{y}_C, z_C, \dot{G}_n, \dot{Y}_n, V_1, V_2, V_3)$.

(2) If V_1, V_2 and V_3 are correct, the bank sends the customer $\tilde{v}_C = \tilde{y}_C^{1/e_2} \bmod n$ and charges the customer's account the amount w.

(3) The customer computes $v_C = \tilde{v}_C / r_1 \bmod n$ to obtain the coin (x_C, v_C), where $v_C \equiv (f_1 x_C^{e_1} + f_2)^{1/e_2} \pmod{n}$.

5.2.3 Payment

Consider a binary tree with ℓ levels. Let n_{j_1} $(j_1 = 0)$ denote the root node, and let $n_{j_1 \cdots j_u}$ denote the left (resp., right) child of the parent node $n_{j_1 \cdots j_{u-1}}$ if $j_u = 0$ (resp., $j_u = 1$), for $u = 2, \ldots, \ell$. In the payment protocol, the customer pays the merchant any amount $\tilde{w}$ $(\leq w = 2^{\ell - 1})$. Let $[\tilde{w}_\ell \cdots \tilde{w}_1]$ be the binary representation of $\tilde{w}$. Then, if $\tilde{w}_{\ell - u + 1} = 1$ $(1 \leq u \leq \ell)$, the customer pays a node $n_{j_1 \cdots j_u}$ among the nodes in the u-th level that have not been previously paid and do not have the ancestor-descendant relationship with the previously paid nodes. Here, the payment protocol for a node $n_{j_1 \cdots j_u}$ is shown. By executing this payment protocol for multiple nodes parallel, the payment for any amount is accomplished. Let m be the concatenation of the identifier of the merchant obtaining the payment, the time when the payment is made, and the location of the currently paid node in the tree. To detect over-spending of the node, an F value of the paid node is used, as well as the previous system. The F value of the root node, denoted F_{j_1}, is $\tilde{h}^{y_C}$. The F value of a node $n_{j_1 \cdots j_u}$, denoted $F_{j_1 \cdots j_u}$, is $h_{(u, j_u)}^{F_{j_1 \cdots j_{u-1}}}$ where $F_{j_1 \cdots j_{u-1}}$ is the F value of the parent node. Note that, in the proposed system, the F value of the paid node is computed by the trustees after the deposit. The detail payment protocol is as follows:

(1) The customer computes $\tilde{G}_n = g_n^{\tilde{r}}$ and $\tilde{Y}_n = y_n^{\tilde{r}} h^{y_C}$ for $\tilde{r} \in_R Z_n^*$, and computes $\dot{z}_n = \tilde{g}_n^{y_C}$ and $\tilde{z}_{p_2} = \tilde{g}_{p_2}^{\tilde{h}^{y_C}}$ for $\tilde{g}_n \in_R \mathcal{G}_n$ and $\tilde{g}_{p_2} \in_R \mathcal{G}_{p_2}$. In addition, the customer computes the following SPKs:

$$\tilde{V}_1 = \mathrm{SPK}\{(\alpha, \beta) : \tilde{Y}_n = y_n^{\alpha} h^{\beta^{e_1}}\}(m),$$
$$\tilde{V}_2 = \mathrm{SPK}\{(\gamma, \delta) : \tilde{Y}_n^{f_1} h^{f_2} = y_n^{\gamma} h^{\delta^{e_2}}\}(m),$$
$$\tilde{V}_3 = \mathrm{SPK}\{(\epsilon, \zeta) : \tilde{G}_n = g_n^{\epsilon} \wedge \tilde{Y}_n = y_n^{\epsilon} h^{\zeta} \wedge \dot{z}_n = \tilde{g}_n^{\zeta}\}(m),$$
$$\tilde{V}_4 = \mathrm{SPK}\{(\eta) : \dot{z}_n = \tilde{g}_n^{\eta} \wedge \tilde{z}_{p_2} = \tilde{g}_{p_2}^{\tilde{h}^{\eta}}\}(m).$$

Note that these are the same as the previous. Furthermore, instead of computing the F value of the paid node with the verifiability, the customer computes a ciphertext of the root F value, $G_n = g_n^r$ and $Y_n = y_n^r \tilde{h}^{y_C}$, for $r \in_R Z_n^*$. The customer proves the correctness by

$$\tilde{V}_5 = \mathrm{SPK}\{(\iota, \kappa, \lambda) : G_n = g_n^{\iota} \wedge Y_n = y_n^{\iota} \tilde{h}^{\kappa} \wedge \tilde{Y}_n = y_n^{\lambda} h^{\kappa}\}(m).$$

Finally, the customer sends the merchant $A = (\tilde{G}_n, \tilde{Y}_n, \tilde{g}_n, \dot{z}_n, \tilde{g}_{p_2}, \tilde{z}_{p_2}, G_n, Y_n, \tilde{V}_1, \tilde{V}_2, \tilde{V}_3, \tilde{V}_4, \tilde{V}_5)$.

(2) The merchant verifies that A is correctly formed. If the merchant is successful, this payment is permitted.

Remark 2 In the previous protocol, in addition to $(\tilde{G}_n, \tilde{Y}_n, \tilde{g}_n, \dot{z}_n, \tilde{g}_{p_2}, \tilde{z}_{p_2}, \tilde{V}_1, \tilde{V}_2, \tilde{V}_3, \tilde{V}_4)$, the customer sends the merchant the F value $F_{j_1 \cdots j_u}$ of the paid node, after computing all the F values $F_{j_1}, \ldots, F_{j_1 \cdots j_{u-1}}$ of intermediate nodes from the root to the target, such as $F_{j_1} = \tilde{h}^{y_C}$, $F_{j_1 j_2} = h_{(2, j_2)}^{F_{j_1}}, \ldots, F_{j_1 \cdots j_u} = h_{(u, j_u)}^{F_{j_1 \cdots j_{u-1}}}$. Furthermore, he sends the corresponding commitments $\tilde{F}_1 = \dot{h}_1^{y_C}, \tilde{F}_2 = \dot{h}_2^{F_{j_1}}, \ldots, \tilde{F}_{u-1} = \dot{h}_{u-1}^{F_{j_1 \cdots j_{u-2}}}$ for randomly chosen bases $\dot{h}_1, \ldots, \dot{h}_{u-2}$, and confirms the merchant the correctness of $F_{j_1 \cdots j_u}$ by sending the SPKs to prove the knowledge of DLs and double DLs:

$$\log_{\dot{h}_1} \tilde{F}_1 = \log_{\tilde{h}} (\log_{\dot{h}_2} \tilde{F}_2),$$
$$\ldots,$$
$$\log_{\dot{h}_{u-1}} \tilde{F}_{u-1} = \log_{h_{(u-1, j_{u-1})}} (\log_{h_{(u, j_u)}} F_{j_1 \cdots j_u}).$$

In the proposed one, these are replaced by $(G_n, Y_n, \tilde{V}_5)$ only.

5.2.4 Deposit

(1) The merchant sends the bank the transcript of the payment A.

(2) The bank verifies that the transcript is correctly formed. Only if it is correctly formed, the bank permits the payment to deposit the paid amount in the merchant's account. After that, to detect over-spending, the bank sends the trustees the transcript.

(3) The trustees verify the correctness of the transcript. Only if the quorum of the trustees agree on the correctness, they cooperatively conduct the following:

(a) Let $n_{j_1 \cdots j_u}$ be the paid node in the level u. Note that the ElGamal ciphertext (G_n, Y_n) in the transcript should include the plaintext F_{j_1}, that is the root F value. Then, the quorum of trustees cooperatively computes the ciphertext of the F value of the paid node. In case of the root, skip this substep. Otherwise, they perform the PCOWF for (G_n, Y_n) w.r.t. the one-way function $f_{(2,j_2)}(a) = h^a_{(2,j_2)}$ to obtain a ciphertext (G_{p_2}, Y_{p_2}). This ciphertext should include the plaintext $F_{j_1 j_2} = h^{F_{j_1}}_{(2,j_2)}$. Up to the targeted ciphertext, they continue to perform the PCOWF w.r.t. the one-way functions $f_{(3,j_3)}(a) = h^a_{(3,j_3)}, \ldots, f_{(u,j_u)}(a) = h^a_{(u,j_u)}$. Let (G_{p_u}, Y_{p_u}) be the targeted ciphertext including the paid F value $F_{j_1 \cdots j_u}$ as the plaintext.

(b) The quorum of the trustees performs the verifiable threshold decryption for (G_{p_u}, Y_{p_u}) to obtain $F_{j_1 \cdots j_u}$. They return the bank $F_{j_1 \cdots j_u}$ together with the SPKs in the whole computations in these substeps (a) and (b).

(4) The bank uses $F_{j_1 \cdots j_u}$ to detect whether the corresponding node $n_{j_1 \cdots j_u}$ were overspent, as well as the previous system. If the same node is used, the sameness of F value indicates over-spending. If the nodes $n_{j_1 \cdots j_u}$ and $n_{j_1 \cdots j_{u'}}$ $(u < u')$ with the ancestor-descendant relationship are used which also means overspending, the corresponding $F_{j_1 \cdots j_u}$ and $F_{j_1 \cdots j_{u'}}$ have relations as $F_{j_1 \cdots j_{u+1}} = h^{F_{j_1 \cdots j_u}}_{(u+1, j_{u+1})}, \ldots, F_{j_1 \cdots j_{u'}} = h^{F_{j_1 \cdots j_{u'-1}}}_{(u', j_{u'})}$ for F values of the intermediate nodes, $F_{j_1 \cdots j_{u+1}}, \ldots, F_{j_1 \cdots j_{u'-1}}$. Thus, the relation enables the bank to detect overspending. For the detail, refer to the paper[9]. If over-spending occurs, the bank traces the over-spender by using the owner tracing for the corresponding payment transcript.

Remark 3 In the previous protocol, since the bank directly obtains the F value $F_{j_1 \cdots j_u}$ of the paid node from the payment transcript, above Step (3) is not conducted.

5.2.5 Tracing

This is the same as the previous. For owner tracing, the trustees decrypt the ciphertext $(\tilde{G}_n, \tilde{Y}_n)$ in the payment to obtain $z_C = h^{y_C}$, which matches to the corresponding withdrawal protocol. For coin tracing, the trustees decrypt the ciphertext $(\dot{G}_n, \dot{Y}_n)$ in the withdrawal to obtain $\tilde{h}^{y_C}$. The merchant and bank can find the matched $\tilde{z}_{p_2} = \tilde{g}_{p_2}^{\tilde{h}^{y_C}}$ in each payment.

5.3 Security Consideration

As the anonymity revocation, divisibility and off-line-ness are straightforward, the remaining properties are discussed.

Unforgeability: As well as the previous, this holds because the coin (x_C, v_C) is unforgeable and SPK in the payment shows the knowledge of the coin.

No over-spending: It is assured that (G_n, Y_n) in the payment is an ElGamal ciphertext of $F_{j_1} = \tilde{h}^{y_C}$, because of the SPKs. In the deposit, if the trustees obey the protocol, the trustees compute the target F values from (G_n, Y_n), since the PCOWF protocol and the threshold decryption satisfy the completeness. Furthermore, each computation can be verified. Thus, the bank obtains the correct F values of the paid nodes, and can detect over-spending. The over-spender can be traced by the owner tracing.

No swindling: This depends on the secrecy of x_C. Since the original part does not reveal x_C, consider the newly added parts. In the payment, (G_n, Y_n) and $\tilde{V}_5$ are added. They are a ciphertext and SPK, and thus do not reveal x_C. In the deposit, the PCOWF and verifiable decryption protocols are added. They do not also reveal any information about x_C.

Unlinkability and anonymity: We discuss that the newly added parts have no influence. As mentioned in no swindling, the ciphertext and SPK in the payment has no information. In the deposit, the PCOWF and verifiable decryption protocols reveal only the target F value, which is revealed in the previous system. Therefore, these properties hold.

From the above discussion, the security of the proposed system is the same as that of the previous. The unlinkability and anonymity depend

Table 2 Worst approximate number of exponentiations required in payment and deposit protocols in case of coin with value 100,000.

	Payment (Customer or Merchant)	Deposit (Trustee)
Ref. 9)	600	0
Ours	80	600

on the trust of the trustees in the threshold setting. Note that the trust is also the same among both systems, as mentioned in Section 2

5.4 Efficiency Consideration

The payment protocol is very efficient, as follows. The original part $(\tilde{G}_n, \tilde{Y}_n, \tilde{g}_n, \dot{z}_n, \tilde{g}_{p_2}, \tilde{z}_{p_2}, \tilde{V}_1, \tilde{V}_2, \tilde{V}_3, \tilde{V}_4)$ needs about 70 exponentiations, by the estimation in the paper[9]. The newly added (G_n, Y_n) and $\tilde{V}_5$ need less than 10 exponentiations. The previous payment protocol needs about 600 exponentiations in total for the worst, if the monetary amount is about 100,000.

On the other hand, in the deposit protocol, the trustees have to compute the F values of the paid nodes from the root via the ElGamal ciphertexts. The dominant cost is the ZPKs proving that the double DL equals the DL. These costs are comparable to those of the previous payment protocol. Thus, in the same situation, a trustee computes about 600 exponentiations. These discussions are summarized in **Table 2**.

Note that the trustees' vast computations must be executed sequentially. Thus, the computation time is in proportion to the number of trustees, N. Furthermore, this cost is required for all the payment transcripts. This may cause the problem on the scalability. However, the trustees can have the more powerful computation ability than the users. In addition, these computations can be executed by multiple groups of trustees, where each group consists of trustees with the threshold trust. Therefore, the proposed system has the sufficient reality.

6. Conclusion

We have proposed an improved unlinkable divisible e-cash system, where the payment protocol is more efficient.

In the propose system (also the previous system), to check over-spending, the bank has to check the chain of one-way functions for all paid nodes of all the payments, which is vast. Thus, a further work is to reduce the vast computations of checking over-spending. This may be accomplished by the payer's computing online with the bank.

References

1) Chaum, D.: Blind Signatures for Untraceable Payments, *Advances in Cryptology: Proc. CRYPTO'82*, pp.199–203, Plenum Press (1983).

2) Okamoto, T. and Ohta, K.: Universal Electronic Cash, *Advances in Cryptology — CRYPTO'91*, LNCS 576, pp.324–337, Springer-Verlag (1992).

3) Brands, S.: Untraceable off-line cash in wallets with observers, *Advances in Cryptology — CRYPTO'93*, LNCS 773, pp.302–318, Springer-Verlag (1994).

4) Okamoto, T.: An Efficient Divisible Electronic Cash Scheme, *Advances in Cryptology — CRYPTO'95*, LNCS 963, pp.438–451, Springer-Verlag (1995).

5) Davida, G., Frankel, Y., Tsiounis, Y. and Yung, M.: Anonymity Control in E-Cash Systems, *Proc. First Financial Cryptography Conference* (*FC'97*), LNCS 1318, pp.1–16, Springer-Verlag (1997).

6) Chan, A., Frankel, Y. and Tsiounis, Y.: Easy Come-Easy Go Divisible Cash, *Advances in Cryptology — EUROCRYPT'98*, LNCS 1403, pp.561–575, Springer-Verlag (1998).

7) Fujisaki, E. and Okamoto, T.: Practical Escrow Cash Schemes, *IEICE Trans. Fundamentals*, Vol.E81-A, No.1, pp.11–19 (1998).

8) Nakanishi, T. and Sugiyama, Y.: Group Signature Scheme with Signature Tracing and Its Application to an Electronic Coupon System, *Trans. IPS Japan*, Vol.42, No.8, pp.2030–2039 (2001).

9) Nakanishi, T. and Sugiyama, Y.: An Efficiency Improvement on an Unlinkable Divisible Electronic Cash System, *IEICE Trans. Fundamentals*, Vol.E85-A, No.10, pp.2326–2335 (2002).

10) Pfitzmann, B. and Waidner, M.: How to Break and Repair a "Provably Secure" Untraceable Payment System, *Advances in Cryptology — CRYPTO'91*, LNCS 576, pp.338–350, Springer-Verlag (1992).

11) Camenisch, J. and Stadler, M.: Efficient Group Signature Schemes for Large Groups, *Advances in Cryptology — CRYPTO'97*, LNCS 1294, pp.410–424, Springer-Verlag (1997).

12) Nakanishi, T. and Sugiyama, Y.: An Unlinkable Divisible Electronic Cash with User's Less Computations Using Active Trustees, *Proc. 2002 International Symposium on Information Theory and its Applications* (*ISITA2002*), pp.547–550 (2002).

13) Jakobsson, M. and Juels, A.: Mix and Match: Secure Function Evaluation via Ciphertexts, *Advances in Cryptography — ASI-*

ACRYPT2000, LNCS 1976, pp.162–177, Springer-Verlag (2000).

14) von Solms, S. and Naccache, D.: On Blind Signatures and Perfect Crimes, *Computers and Security*, Vol.11, No.6, pp.581–583 (1992).

15) Fiat, A. and Shamir, A.: How To Prove Yourself: Practical Solutions to Identification and Signature Problems, *Advances in Cryptology — CRYPTO'86*, LNCS 263, pp.186–194, Springer-Verlag (1987).

16) Bellare, M. and Rogaway, P.: Random Oracles are Practical: A Paradigm for Designing Efficient Protocols, *Proc. First Annual Conference on Computer and Communications Security*, pp.62–73, Association for Computing Machinery (1993).

17) Desmedt, Y. and Frankel, Y.: Threshold Cryptosystems, *Advances in Cryptology — CRYPTO'89*, LNCS 435, pp.307–315, Springer-Verlag (1990).

18) Pedersen, T.P.: A Threshold Cryptosystem without a Trusted Party, *Advances in Cryptology — EUROCRYPT'91*, LNCS 547, pp.522–526, Springer-Verlag (1991).

19) Gennaro, R., Jarecki, S., Krawczyk, H. and Rabin, T.: Secure Distributed Key Generation for Discrete-Log Based Cryptosystems, *Advances in Cryptology — EUROCRYPT'99*, LNCS 1592, pp.295–310, Springer-Verlag (1999).

20) Abe, M.: Universally Verifiable Mix-net with Verification Work Independent of the Number of Mix-Servers, *IEICE Trans. Fundamentals*, Vol.E83-A, No.7, pp.1431–1440 (2000).

21) Gennaro, R., Jarecki, S., Krawczyk, H. and Rabin, T.: Robust Threshold DSS Signatures, *Advances in Cryptology — EUROCRYPT'96*, LNCS 1070, pp.354–371, Springer-Verlag (1996).

(Received November 27, 2002)
(Accepted June 3, 2003)

Toru Nakanishi was born in Kagawa, Japan, on May 22, 1971. He received the M.E. and Ph.D. degrees in information and computer sciences from Osaka University, Toyonaka, Osaka, Japan, in 1995 and 2000, respectively. He joined the Department of Information Technology at Okayama University, Japan, as a research associate in 1998, and moved to the Department of Communication Network Engineering in 2000. His research interests are cryptography and information security. He is a member of IEICE.

Yuji Sugiyama was born in Okayama, Japan, on May 20, 1951. He received the B.E., M.E. and Ph.D. degrees in information and computer sciences from Osaka University, Toyonaka, Osaka, Japan, in 1974, 1976 and 1983, respectively. He joined the faculty of Osaka University in 1977. Since 2000, he has been a professor in the Department of Communication Network Engineering at Okayama University. His current research interests include algebraic specifications and implementation of algebraic languages. He is a member of IEICE.