Regular Paper

Anonymous Statistical Survey of Attributes Using Distributed Plaintext Membership Test

Toru Nakanishi† and Yuji Sugiyama†

A distributor of digital contents desires to collect users' attributes. This is because the distributor can grasp the image of users, and work out the marketing strategy. On the other hand, the users do not desire to offer the attributes owing to the privacy protection. For anonymous surveys, a protocol to generate statistical results of the attributes is previously proposed, where the extra information is not released beyond the statistical results. However, in the simple application of this protocol to the surveys, the correctness of the statistical results is not assured, since the distributor cannot verify whether the users offer the correct attributes. In this paper, under the assumption that some trusted third parties exist, an anonymous statistical survey system of attributes where the distributor can verify the correctness is proposed. In this accomplishment, a new cryptographic distributed protocol is introduced, which is called the distributed plaintext membership test protocol.

1. Introduction

1.1 Backgrounds and Contributions

Recently, digital contents have been distributed on the computer network for the commercial purposes, where the distributor sends users the digital contents including texts, images and sound, while the users watch advertisements or pay the distributor the money. It is desirable that these services are conducted anonymously, since otherwise the distributor can collect the history that indicates which contents a user utilizes and furthermore the distributor may leak the history to others. By contrast, the distributor wants to grasp the image of users, since the distributor can work out the strategy according to the image. This may also benefit the users, since they may obtain the more suitable contents. One method to grasp the image is to collect the attributes of the users, which are concretely the gender, age, job and so on. However, even if offering the attributes during the services is conducted anonymously, offering many attributes may help the distributor to trace the identity of the user. Only the statistical results of the attributes may give the distributor useful information to grasp the image of the entire users. The statistical result for an attribute type means the list of pairs (attribute value, the number of users with the value). For example, for the attribute type gender, this is ((male, 60), (female, 40)).

This paper☆ proposes an anonymous statistical survey system of attributes. In this system, a user obtains the certificate on his attributes from a trusted third party (TTP), when registering with the system. Then, during an anonymous service, the user offers the distributor his attribute values as commitments, whose validity is assured by the certificates. The distributor collects the commitments, and requests other TTPs to generate the statistical results of attributes. In the proposed system, the anonymity and verifiability are satisfied. The anonymity means that the offering protocol is anonymously conducted, and that the TTPs' generating protocol does not reveal any useful information to identify the user. The verifiability means that the distributor can verify the correctness of the statistical results. In this accomplishment, a new cryptographic distributed protocol is introduced, which is called the distributed plaintext membership test protocol.

1.2 Related Works

Sako proposes a protocol executed among several TTPs in order to generate statistical results of attributes for anonymous survey systems [3]. The merit of this protocol is that it releases no extra information beyond the statistical results. Each TTP is in charge of each attribute type, and the TTP obtains only the information of the corresponding attribute type. The TTP's input is the set of the ciphertexts encrypted with the TTP's public key from at-

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

☆ Preliminary versions of the paper appeared in CSS2000 [1] and ACISP2001 [2].

tribute values on the corresponding attribute type. The output is only the statistical result of the attribute type. By using this protocol, the anonymous survey system of attributes is simply constructed as follows: A user sends the distributor the ciphertexts of the user's attribute values, and the distributor collects them. After collecting a certain amount of ciphertexts, the distributor gives the TTPs them to execute this protocol. Then, the distributor can obtain the statistical results of all attribute types without the extra information. This protocol has a mechanism to detect a TTP that does not obey the protocol, and thus the distributor can verify that the results are correct, if the distributor can verify that the inputs are correct, that is, the attribute values in the ciphertexts are correct. However, in the above simple survey system, the distributor cannot verify that the user sends the correct attribute values. In the proposed system, owing to the certificate on the attributes and the zero-knowledge proof of knowledge of the certificate, the distributor can verify that the inputs are correct. Furthermore, the generating protocol is also verifiable, and thus so is our entire system. Note that our system adopts the trust model of threshold type, which means that our system is anonymous for even each TTP, unless a quorum of the trustees is corrupted.

A plausible solution is for the user to send the distributor a ciphertext of his attribute value itself in the offering protocol, while proving that the plaintext of the sent ciphertext belongs to the list of the valid attribute values. This proof can be accomplished in the zero-knowledge fashion [4),5)]. The generating protocol simply consists of the shuffle and decryption of the ciphertexts. However, since the user can send a ciphertext of the attribute value that is different from the genuine, this solution does not also satisfy the verifiability. Therefore, as well as our system, the certificate of the attribute value is needed.

1.3 Organization

This paper is organized as follows: Section 2 describes a model and requirements of the anonymous statistical survey system of attributes. Next, as the cryptographic tools used in the proposed system, a group signature scheme, distributed shuffle protocol and a distributed plaintext membership test protocol are introduced in Section 3. Then, a construction and the security of the system are shown in Section 4. In Section 5, since the distributed plaintext membership test protocol is new concept, a construction and the security of the protocol are shown. Finally, Section 6 concludes this paper.

2. A Model and Requirements

The participants in an anonymous statistical survey system of attributes are an attribute authority, users, a distributor, and trustees. The attribute authority is a TTP, and the authority assures the correspondence between the user's genuine attribute values and the encrypted values which are offered from the user. It is assumed that the authority can be convinced of the user's genuine attribute values, and that the authority assures the correct correspondence between the attribute values and the encrypted values. The trustees are also TTPs, and it is assumed that a quorum of them is not corrupted.

The survey system consists of the setup, registration, offering, and generating protocols. In the setup protocol, the secret and public keys of the attribute authority and the trustees are set up. In the registration protocol, a user generates his secret key and public key, and is issued the attribute certificate for a registered value from the attribute authority. In the offering protocol that is executed during the distributor's service in practice, a user sends the distributor the encrypted values correspondent with user's attribute values, whose validity is assured by the attribute certificate. In the generating protocol, given the encrypted values of many users, a quorum of the trustees outputs the statistical result of every attribute type.

These protocols except the offering protocol use an authenticated channel (e.g., by digital signatures), and the offering protocol uses an anonymous channel.

The requirements of anonymous survey system of attributes are as follows:

Correctness: The last output from the generating protocol is the correct statistical result if the participants obey the protocols (Completeness). If a participant disobeys the protocols, it can be detected (Verifiability). Note that the correct statistical result means that the correspondence between each attribute value a and the number of the offering users with a as the genuine attribute value.

Anonymity: The offering protocol is conducted anonymously. That is, the other

party can not identify the user from a transcript of this protocol, and can not also link two transcripts w.r.t. the sameness of the user. Furthermore, the generating protocol does not reveal any useful information to trace the user's identity from the the transcript of the offering protocol.

Note that our definition of the anonymity allows the generating protocol to reveal the information beyond the statistical result, unless the revealed information is influential in identifying users.

3. Building Blocks

To construct a proposed survey system, we use a group signature scheme, a distributed shuffle, and a distributed plaintext membership test protocols. Since only the distributed plaintext membership test protocol is a new concept, we show the concrete constructions in Section 5.

3.1 Group Signature Scheme

The group signature scheme allows a group member to anonymously sign on group's behalf. Furthermore, the anonymity of the signature can be revoked by only a TTP. The participants of this scheme are group members, a group manager, and a revocation manager. The group manager has the authority to decide whether a user belongs to the group, and the revocation manager has the authority to revoke the anonymity of signatures. The group signature scheme can be extended into one with multiple revocation managers, where only a quorum of them can revoke the anonymity. Since we adopt such a scheme, the following definition of the security requirements is based on the model.

Definition 1 A secure group signature scheme satisfies the following properties:

Unforgeability: Only group members can sign messages.

Anonymity: It is neither feasible to decide which member signed a message, nor to decide whether two signatures were made by the same signer.

No framing: Neither a group member nor the group manager can sign messages on behalf of other members.

Revocability of anonymity: The anonymity of a signature can be revoked only by a quorum of revocation managers and when necessary. □

We adopt Chamenisch and Stadler's scheme[6]. This uses an ElGamal ciphertext for a registered value, which can be easily extended into one using the threshold ElGamal ciphertext. In threshold encryptions[7], only a quorum of the designated parties can decrypt the ciphertext w.r.t. the parties' public key. The extended group signature scheme is as follows:

Setup protocol: The group manager sets up public and secret keys on a digital signature scheme, and the revocation managers set up public and secret keys on a threshold ElGamal encryption scheme.

Registration protocol: Let f be the one-way function of the form $f(x) = h^x$, where h is an element of a cyclic group with order n (n is an RSA modulus) and x is an element of $\mathcal{Z}_n^*$. When a user wants to participate in the group, the user secretly chooses a random element x, and sends the group manager $f(x)$ together with his identity. Then, the manager returns his digital signature on $f(x)$, denoted as $DS(f(x))$, as the membership certificate.

Signing and verification protocol: As the group signature on a message m, a group member computes $d = Enc(f(x))$ and $p = SPK\{(\alpha, \beta) : d = Enc(f(\alpha)) \wedge \beta = DS(f(\alpha))\}(m)$, where Enc is the ElGamal encryption with the revocation managers' public key. The SPK is the signature converted by the so-called Fiat-Shamir heuristic from a zero-knowledge proof of knowledge. The proof proves secret knowledge α and β satisfying $d = Enc(f(\alpha)) \wedge \beta = DS(f(\alpha))$.
Its verification is accomplished by verifying the SPK.

Anonymity revocation protocol: When the anonymity of a signature (d, p) is revoked, the quorum of the revocation managers cooperatively decrypts d to obtain $f(x)$. Through the registration transcript including $f(x)$, the identity of the signer is found.

For the concrete construction, refer to the paper[6].

3.2 Distributed Shuffle Protocol

We adopt a distributed shuffle protocol on the threshold ElGamal ciphertexts. In the shuffle protocol, the participants are multiple servers, who set up the secret keys and public key on the threshold ElGamal encryption. Given a list of ElGamal ciphertexts $(c_1, \ldots, c_N)$ w.r.t. the public key, the servers cooperatively output a list of permuted ciphertexts $(c'_1, \ldots, c'_N)$ satis-

fying $Dec(c_j) = Dec(c'_{\pi(j)})$ for all j, where Dec is the decryption function and π is a permutation. The requirements of the distributed shuffle protocol are as follows:

Definition 2 A secure distributed shuffle protocol satisfies the following properties:

Correctness: The shuffled result is correct if the servers obey the protocols. If a server disobeys the protocols, it can be detected.

Unlinkability: It is infeasible to determine $\pi(j)$ for any j with non-negligibly better probability unless the quorum of servers cooperates. □

Abe proposes a distributed shuffle protocol on the threshold ElGamal encryption for constructing the Mix-net [8].

3.3 Distributed Plaintext Membership Test Protocol

We introduce a cryptographic tool, called a distributed plaintext membership test protocol. The participants are the same servers as ones in the shuffle protocol. The inputs are (1) lists of plaintexts, and (2) ElGamal ciphertexts w.r.t. the servers' public key. Let the lists denote $L_1, \ldots, L_K$ and let the ciphertexts denote $c_1, \ldots, c_N$. In this paper, assume that the underlying plaintext of each input ciphertext always belongs to a single list in the inputs. Our application to the statistical survey system satisfies this condition. Then, the servers cooperatively output the ID's of the lists, $\tilde{L}_1, \ldots, \tilde{L}_N$ ($\tilde{L}_i \in \{L_1, \ldots, L_K\}$), where the plaintext of input ciphertext c_i belongs to the output list $\tilde{L}_i \in \{L_1, \ldots, L_K\}$. The requirements of the plaintext membership test protocol are as follows:

Definition 3 A secure distributed plaintext membership test protocol satisfies the following properties:

Correctness: The membership results are correct if the servers obey the protocols. If a server disobeys the protocols, it can be detected.

Unlinkability: It is infeasible to link the each input ElGamal ciphertext to the corresponding plaintext in the input lists, unless the quorum of servers cooperates. □

Note that this definition allows anyone to link a ciphertext in the inputs to another ciphertext w.r.t. the sameness of the encrypted plaintexts.

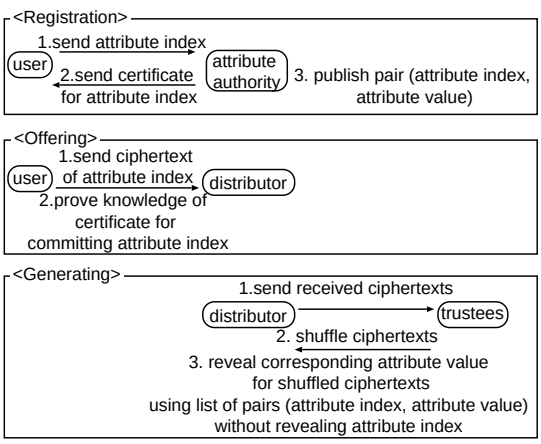

Fig. 1 Overview of proposed system.

4. An Anonymous Statistical Survey System of Attributes

4.1 Overview

Before describing the detail of the proposed system, we show the overview. See **Fig. 1**.

In the registration, a user sends the attribute authority an attribute index. The index is a proper value to each user, and is correspondent with his genuine attribute value. In return, the authority issues the user the certificate for the attribute index. The certificate assures the correspondence between the index and attribute value. The correspondence is published.

When offering his attribute value, the user sends a distributor a ciphertext of the attribute index by using the trustees' public key. Then, the user should prove the correctness of the index. This can be accomplished by sending the certificate. However, this simple way disturbs the unlinkability of the offering transcripts in the requirement of the anonymity. Therefore, in our system, the user proves the knowledge of the certificate in a zero-knowledge fashion, which is the technique used in the group signature scheme.

For generating the statistical result, the distributor sends the trustees the ciphertexts that are received from users. At first, the trustees cooperatively shuffle the ciphertexts. Next, for each shuffled ciphertext, the trustees cooperatively reveal the attribute value that is correspondent with the attribute index encrypted in the ciphertext. The simple way is to decrypt the ciphertext. However, this decryption allows the attribute authority to identify the user from the attribute index, which compromises

the anonymity. In our system, to reveal the attribute value is accomplished without revealing the attribute index. By counting the revealed attribute values, the distributor can generate the statistical result.

Remark:

A plausible solution is for the user to send the distributor a ciphertext of his attribute index, while proving that the plaintext of the sent ciphertext belongs to the list of all the valid attribute indices by the techniques [4),5)]. However, since the computation complexity is proportional to the number of all the registering users, it is inefficient and impractical for users using a variety of computers. Our solution is for the user to prove the knowledge of the certificate for the attribute index. This is derived from the group signature scheme [6)]. The complexity does not depend on the number of all the registering users.

4.2 Proposed System

Using the group signature scheme, distributed shuffle protocol, and distributed plaintext membership test protocol, we construct an anonymous statistical survey system of attributes. In the application, the users play the roles of the group members, the distributor plays the role of the verifier, the attribute authority plays the role of the group manager, and the trustees play the roles of both the revocation managers and the servers. Consider the case of single attribute type. The cases of multiple attribute types are straightforward. Each protocols are as follows:

Setup protocol: The setup protocol of the group signature scheme is conducted. Note that the trustees cooperatively set up keys of the threshold ElGamal encryption.

Registration protocol: The registration protocol of the group signature scheme is conducted, where the attribute authority preserves the attribute value of the registering user instead of the identity. Thus, since each $y = f(x)$ is correspondent with each attribute value, y is the attribute index. The lists of the indices of all users are made public, denoted as $Y_1, \ldots, Y_K$ (K is the number of the lists), where each list Y_j consists of indices with the same attribute values. The correspondence between the lists and the corresponding attribute values is also public. Furthermore, the membership certificate plays the role of the attribute certificate.

Offering protocol: To offer the attribute value, the user sends the distributor the user's group signature on a random message chosen by the distributor. The distributor can verify the correctness of the index by verifying the signature. The distributor collects signatures of users.

Generating protocol: To obtain the statistical result of the attribute type, the distributor sends the trustees the received signatures. Each signature includes the ElGamal ciphertext of the attribute index w.r.t. the trustees' public key. Let all the ciphertexts be $c_1, \ldots, c_N$, where N is the number of the ciphertexts. At first, the trustees cooperatively shuffle the list $L = (c_1, \ldots, c_N)$ to obtain $\tilde{L} = (\tilde{c}_1, \ldots, \tilde{c}_N)$. After that, given the lists $Y_1, \ldots, Y_K$ and shuffled ciphertexts $\tilde{c}_1, \ldots, \tilde{c}_N$, the trustees cooperatively conduct the distributed plaintext membership test to output the ID's of lists Y_j such that the plaintext of each $\tilde{c}_i$ belongs to each Y_j. Each list Y_j indicates the attribute value. By counting the attribute values made correspondent with all ciphertexts, the statistical result is computed.

Remark:

Sako's generating protocol [3)] can output the multi-cross table of attributes. The multi-cross table means one showing the correlation among more than one attribute types. For example, a multi-cross table of the gender and occupation shows the number of Male and employed people, Male and unemployed people, and so on. Our system can also generate this type of statistical results, as follows: Consider two attribute types for simplicity. The registration is independently conducted for both attribute types. When offering attribute values, a user conducts the offering protocol for both types, where a pair of two ciphertexts for both attribute types are offered. In generating protocol, the distributor sends a list of received pairs $((c_1, \tilde{c}_1), \ldots, (c_N, \tilde{c}_N))$, and the servers shuffle the list $((c_1, \tilde{c}_1), \ldots, (c_N, \tilde{c}_N))$, where the output is a list of permuted pairs of randomized ciphertexts, $((c'_1, \tilde{c'}1), \ldots, (c'_N, \tilde{c'}N))$. Thus, underlying plaintexts within each input pair are the same as those within the permuted output pair. Note that this type of shuffle protocol can be easily constructed from some shuffle protocols such as Abe's one [8)]. After the shuffle, all the shuffled ciphertexts are made correspondent

to the attribute values, by the distributed plaintext membership test protocol. This produces the number of pairs of any values for both attribute types, which means the multi-cross table.

4.3 Security

Now, we discuss that the proposed system satisfies the requirements in Section 2.

Correctness: Owing to the soundness of the SPK p in the group signature and the protection of the replay attack that is brought by the use of the random message m, it is assured that the user knows the attribute certificate $DS(f(x))$ such that the ciphertext is encrypted from $y = f(x)$. Owing to the unforgeability of the certificate, it is assured that the user, in advance, registered y, which is published as an element in the list correspondent with the genuine attribute value a. Thus, the user can offer only the ciphertext of y that is correspondent with a, or his dishonest acts can be detected. Therefore, the distributor gives the generating protocol the ciphertexts of the attribute indices that are correspondent with the users' correct attribute values. Let the attribute indices be $(y_1, \ldots, y_N)$, and the corresponding lists and the attribute values be $(Y_1, \ldots, Y_N)$ and $(a_1, \ldots, a_N)$, respectively.

The remain are to show that the generating protocol outputs the correct statistical result, which is the number of each attribute value, and to show the correctness can be verified. The ciphertexts $(c_1, \ldots, c_N)$ of the input are shuffled into $(\tilde{c_1}, \ldots, \tilde{c_N})$. Owing to the correctness of the shuffle protocol, the plaintexts of the latter ciphertexts also form $(y_1, \ldots, y_N)$ in different order. The next distributed plaintext membership test protocol outputs the lists of attribute indices. Owing to the correctness of the protocol, they forms the corresponding $(Y_1, \ldots, Y_N)$. As a result, the generating protocol generates the correct attribute values $(a_1, \ldots, a_N)$, and thus counting them produces the correct statistical result. Furthermore, note that the above correctness can be verified by the distributor, owing to the verifiability of the used tools.

Anonymity: In the offering protocol, the group signature on a random message is sent. Thus, the transcript itself is anonymous, that is, it is infeasible to trace the owner's identity, nor to decide the sameness of the owners.

Next, it is shown that the generating protocol reveals no useful information to identify a user. After the shuffle protocol, the shuffled ElGamal ciphertexts do not reveal any information. Next, the plaintext membership test protocol outputs the shuffled lists leading the corresponding attribute values. The shuffled attribute values have no link to the offering protocol transcripts and no link to the published attribute indices. Thus, the values also reveal no information except for the following. As noted after Definition 3, the plaintext membership test protocol allows anyone to link a ciphertext in the inputs to another ciphertext w.r.t. the sameness of the encrypted plaintexts.

In the remain, it is discussed that the link of the ciphertexts w.r.t. the sameness of the encrypted plaintexts is little influential information in identifying the users. Since the plaintext is concretely the attribute index proper to each user, the link leads to the information indicating how many users offer a known attribute value, and indicating how many times each unknown user offers the value. For example, it is the information indicating that two users offer Male as the attribute value, and indicating that one user offers it two times and another offers it five times. This information is statistical, since there is no link from each unknown users to the actual offering protocol. Therefore, we consider that this information is little influential. In our concrete construction of the distributed plaintext membership test protocol, though this information is revealed, the efficiency is gained as the compensation.

5. Constructions of Distributed Plaintext Membership Test Protocol

5.1 Construction by Mix and Match Protocol

A construction of the distributed plaintext membership test protocol is the use of the mix and match protocol [9]. The mix protocol is a variant of shuffle protocols, where, given a list of the plaintexts, the servers output a list of the shuffled ciphertexts. In the match protocol, given two ciphertexts, the distributed servers check whether the corresponding plaintexts are

the same without revealing any other information about the plaintexts. Furthermore, these protocols have the parts proving the correctness. Then, a distributed plaintext membership test protocol can be constructed as follows: The inputs are ciphertexts, c_i $(i = 1, \ldots, N)$, two lists of plaintexts, L, L'. Here, for simplicity, consider the case of two lists. The servers cooperatively shuffle the plaintexts in each L and L'. Then, for each c_i $(i = 1, \ldots, N)$, the servers use the match protocol to check whether the plaintext of c_i is the same as all the shuffled ciphertexts in L and L'. The match indicates the membership lists.

In this construction, the computation cost is $O(NK)$, where K is the total number of the plaintexts in all the lists. In the following, we propose a construction with $O(N + K)$ cost.

5.2 Proposed Construction

The proposed construction is derived from the mix protocol [8]. The mix protocol is also a variant of shuffle protocols, where given ciphertexts are shuffled and furthermore decrypted cooperatively by the servers. In addition, the mix protocol includes the proof parts to confirm the validity of the servers' shuffle and decryption. Our distributed plaintext membership test protocol uses a variant of the shuffle part and its proof part, and replaces the decryption and its proof parts with a match and its proof parts. The replaced parts include the decryption and its proof part. For simplicity, only the case of two lists and one ciphertext as inputs is shown. The extension into cases of more lists is straightforward. The extension into cases of more ciphertexts is shown in Section 5.4.

Here, we deal a variant of ElGamal encryption, since the variant is used by the group signature scheme in the application to the anonymous statistical survey system of attributes. On the variant, we consider the threshold ElGamal encryption as follows: Let g be a generator of a cyclic group with order n that is an RSA modulus, which is required by the group signature scheme. Then, in the threshold ElGamal encryption, a secret key $x \in_U \mathcal{Z}_n^*$ is shared among the servers by a verifiable secret sharing scheme (e.g., Pedersen's scheme [10]), and the corresponding public key is $y = g^x$, where $\in_U$ denotes uniform random selection. Given a plaintext $m \in \langle g \rangle$, the threshold ElGamal ciphertext is $(C_1 = g^r, C_2 = y^r m)$, where $r \in_U \mathcal{Z}_n^*$ is a random factor on the encryption. The decryption is cooperatively executed by servers. In Ref. 8), the concrete construction of the threshold ElGamal encryption is shown, where the decryption outputs the plaintext with no other information, while the correctness is proved by the servers with a zero-knowledge proof of knowledge. Since this construction can be applied to the variant of ElGamal encryption, we construct a distributed plaintext membership test protocol on the threshold ElGamal encryption.

Now, we show a construction of a distributed plaintext membership test protocol, where the shuffle, its proof, match, and its proof parts are executed sequentially.

Shuffle part: In this part, for inputs the lists of plaintexts $L = (m_1, \ldots, m_k)$, $L' = (m'_1, \ldots, m'_{k'})$, and an ElGamal ciphertext (C_1, C_2), the servers $S_1, \ldots, S_\ell$ cooperatively output $\dot{L} = (\dot{m}_1, \ldots, \dot{m}_k)$, $\dot{L}' = (\dot{m}'_1, \ldots, \dot{m}'_{k'})$ and $(\dot{C}_1, \dot{C}_2)$ such that

$$\begin{aligned} \dot{m}_j &= m^t_{\pi(j)} \text{ for all } 1 \le j \le k, \\ \dot{m}'_{j'} &= m'^t_{\pi'(j')} \text{ for all } 1 \le j' \le k', \\ \dot{C}_1 &= C_1^t, \text{and} \\ \dot{C}_2 &= C_2^t, \end{aligned}$$

for random permutations π, π', and $t \in_U \mathcal{Z}_n^*$.

The task of each server S_i is as follows. S_i receives two lists $(m_{i-1,1}, \ldots, m_{i-1,k})$ and $(m'_{i-1,1}, \ldots, m'_{i-1,k'})$, and two values E_{i-1} and F_{i-1}, where $m_{0,1} = m_1, \ldots, m_{0,k} = m_k, m'_{0,1} = m'_1, \ldots, m'_{0,k'} = m'_{k'}, E_0 = C_1$ and $F_0 = C_2$. S_i chooses two random permutations π_i and π'_i and a random factor $t_i \in_U \mathcal{Z}_n^*$. Then, S_i computes

$$\begin{aligned} m_{i,j} &= m^{t_i}_{i-1,\pi_i(j)} \text{ for all } 1 \le j \le k, \\ m'_{i,j'} &= m'^{t_i}_{i-1,\pi'_i(j')} \text{ for all } 1 \le j' \le k', \\ E_i &= E_{i-1}^{t_i}, \text{and} \\ F_i &= F_{i-1}^{t_i}. \end{aligned}$$

S_i's output consists of $(m_{i,1}, \ldots, m_{i,k})$, $(m'_{i,1}, \ldots, m'_{i,k'})$, E_i and F_i. The next server works in the same way, and the process continues up to S_ℓ. The output of this protocol consists of $\dot{L} = (\dot{m}_1 = m_{\ell,1}, \ldots, \dot{m}_k = m_{\ell,k})$, $\dot{L}' = (\dot{m}'_1 = m'_{\ell,1}, \ldots, \dot{m}'_{k'} = m'_{\ell,k'})$ and $(\dot{C}_1 = E_\ell, \dot{C}_2 = F_\ell)$.

For above random permutations π_i and π'_i and factors t_i, $t = \prod_{i=1}^{\ell} t_i$, $\pi = \pi_1 \cdots \pi_\ell$

and $\pi' = \pi'_1 \cdots \pi'_\ell$ should hold.

Shuffle proof part: In this part, the servers $S_1, \ldots, S_\ell$ cooperatively prove a verifier V that they honestly conduct the shuffle part. The servers cooperatively conduct the followings σ times, which indicates the error probability $1/2^\sigma$.

(1) S_i receives $(\tilde{m}_{i-1,1}, \ldots, \tilde{m}_{i-1,k})$ and $(\tilde{m'}_{i-1,1}, \ldots, \tilde{m'}_{i-1,k'})$, and two values $\tilde{E}_{i-1}$ and $\tilde{F}_{i-1}$, where $\tilde{m}_{0,1} = m_1, \ldots, \tilde{m}_{0,k} = m_k, \tilde{m'}_{0,1} = m'_1, \ldots, \tilde{m'}_{0,k'} = m'_{k'}, \tilde{E}_0 = C_1$ and $\tilde{F}_0 = C_2$. S_i chooses two random permutations λ_i and λ'_i and a random factor $s_i \in_U \mathcal{Z}_n^*$. Then, S_i computes

$$\tilde{m}_{i,j} = \tilde{m}_{i-1,\lambda_i(j)}^{s_i} \quad \text{for all } 1 \le j \le k,$$
$$\tilde{m'}_{i,j'} = \tilde{m'}_{i-1,\lambda'_i(j')}^{s_i} \quad \text{for all } 1 \le j' \le k',$$
$$\tilde{E}_i = \tilde{E}_{i-1}^{s_i}, \text{and}$$
$$\tilde{F}_i = \tilde{F}_{i-1}^{s_i}.$$

S_i sends $(\tilde{m}_{i,1}, \ldots, \tilde{m}_{i,k})$, $(\tilde{m'}_{i,1}, \ldots, \tilde{m'}_{i,k'})$, $\tilde{E}_i$ and $\tilde{F}_i$ to the next server S_{i+1}. The last server sends $(\tilde{m}_{\ell,1}, \ldots, \tilde{m}_{\ell,k})$, $(\tilde{m'}_{\ell,1}, \ldots, \tilde{m'}_{\ell,k'})$, $\tilde{E}_\ell$ and $\tilde{F}_\ell$ to V and all servers.

(2) V sends $c \in_U \{0, 1\}$ to all servers.

(3) If $c = 0$, each S_i computes a commitment $b_i = BC(i, \lambda_i, \lambda'_i, s_i)$ and distributes the commitment to V and all servers, where BC is a bit commitment scheme. After all commitments are distributed, each S_i opens his commitment by revealing λ_i, λ'_i and s_i. The last server S_ℓ computes $\lambda = \lambda_1 \cdots \lambda_\ell, \lambda' = \lambda'_1 \cdots \lambda'_\ell$ and $s = \prod_{i=1}^{\ell} s_i \pmod n$. Every server verifies that all commitments, λ, λ' and s are correctly made. If this verification fails, this protocol stops. If $c = 1$, each S_i computes $\varphi_i = \pi_i^{-1}\varphi_{i-1}\lambda_i, \varphi'_i = \pi'^{-1}_i \varphi'_{i-1}\lambda'_i$ and $w_i = w_{i-1}s_i/t_i \pmod n$, where φ_0, φ'_0 are the identity permutations and $w_0 = 1 \pmod n$. The last S_ℓ sends $\varphi = \varphi_\ell, \varphi' = \varphi'_\ell$ and $w = w_\ell$ to V and the other servers.

(4) V and each server verify that, if $c = 0$,

$$\tilde{m}_{\ell,j} = m^s_{\lambda(j)} \quad \text{for all } 1 \le j \le k,$$
$$\tilde{m'}_{\ell,j'} = m'^s_{\lambda'(j')} \quad \text{for all } 1 \le j' \le k',$$
$$\tilde{E}_\ell = C_1^s, \text{and}$$
$$\tilde{F}_\ell = C_2^s,$$

and if $c = 1$,

$$\tilde{m}_{\ell,j} = \dot{m}^w_{\varphi(j)} \quad \text{for all } 1 \le j \le k,$$
$$\tilde{m'}_{\ell,j'} = \dot{m'}^w_{\varphi'(j')} \quad \text{for all } 1 \le j' \le k',$$
$$\tilde{E}_\ell = \dot{C}_1^w, \text{and}$$
$$\tilde{F}_\ell = \dot{C}_2^w.$$

Match part: In this part, for the outputs of the shuffle part $\dot{L} = (\dot{m}_1, \ldots, \dot{m}_k)$, $\dot{L'} = (\dot{m'}_1, \ldots, \dot{m'}_{k'})$, and $(\dot{C}_1, \dot{C}_2)$, the servers output the matched list as follows: By Abe's threshold decryption [8], the servers cooperatively decrypt the ElGamal ciphertext $(\dot{C}_1, \dot{C}_2)$ into the randomized plaintext $\dot{m}$. Finally, the servers search the list including $\dot{m}$ to output the matched list.

Match proof part: The servers prove the correctness of the threshold decryption, by Abe's proof scheme [8].

5.3 Security

Before showing the security, the following lemmas are shown. The first lemma shows the randomness of the shuffle part.

Lemma 1 Given all $m_{i-1,\pi_i(j)}$, $m_{i,j}$, $m'_{i-1,\pi'_i(j')}$ and $m'_{i,j'}$, no adversary can determine $\pi_i(j)$ for any j or $\pi'_i(j')$ for any j' with non-negligibly better probability.

Proof:

Assume an adversary $\mathcal{A}$ who outputs the correct $\pi_i(\bar{j})$ for some $\bar{j}$ with non-negligibly better probability. Then, the following adversary $\tilde{\mathcal{A}}$ can determine the sameness of the discrete logarithms. Without loss of generality, we assume $k = 2$. Given $(M, M_1 = M^\alpha, M_2 = M^\beta, M_3 = M^\gamma)$ with $M \in_U \langle g \rangle$, $\alpha, \beta \in_U \mathcal{Z}_n^*$ and $\gamma = \alpha\beta$ or $\gamma \in_U \mathcal{Z}_n^*$, $\tilde{\mathcal{A}}$ chooses a random permutation $\tilde{\pi}$ on $\{1, 2\}$. Then, for inputs $m_{i-1,\tilde{\pi}(1)} = M$, $m_{i-1,\tilde{\pi}(2)} = M_1$, $m_{i,1} = M_2$, and $m_{i,2} = M_3$, $\mathcal{A}$ is run. As a result, when $\gamma = \alpha\beta$, $\mathcal{A}$ outputs the correct $\tilde{\pi}$ with non-negligibly better probability, since M, M_1 are randomized into M_2, M_3 by random factor β. On the other hand, when $\gamma \neq \alpha\beta$, $\mathcal{A}$ can out-

put the correct one with only negligibly better probability, since the inputs are information-theoretically independent. Therefore, with the non-negligibly better probability, $\tilde{\mathcal{A}}$ can determine the sameness of the discrete logarithms, that is, $\log_M M_1 = \log_{M_2} M_3$. □

The next lemma shows the security of the shuffle proof part. The proof of this lemma is similar to that of the original shuffle proof part [8].

Lemma 2 The shuffle proof part is a honest verifier zero-knowledge proof of knowledge.

Proof:

The completeness holds as follows. In the case of $c = 0$, it is clear that the verification equations are satisfied if servers compute the correct values. In the case of $c = 1$,

$$\tilde{m}_{\ell,j} = m^s_{\lambda(j)},$$

for $\lambda = \lambda_1 \cdots \lambda_\ell$ and $s = \prod_{i=1}^{\ell} s_i \pmod n$. On the other hand, from $\dot{m}_j = m^t_{\pi(j)}$, $\varphi = \pi_\ell^{-1} \cdots \pi_1^{-1} \lambda_1 \cdots \lambda_\ell = \pi^{-1}\lambda$ and $w = \prod_{i=1}^{\ell} s_i/t_i = (\prod_{i=1}^{\ell} s_i)/(\prod_{i=1}^{\ell} t_i) = s/t \pmod n$,

$$\begin{aligned}\dot{m}^w_{\varphi(j)} &= (m^t_{\pi\varphi(j)})^w \\ &= (m^t_{\pi\pi^{-1}\lambda(j)})^{s/t} \\ &= m^s_{\lambda(j)}.\end{aligned}$$

Thus,

$$\tilde{m}_{\ell,j} = \dot{m}^w_{\varphi(j)}.$$

Similarly, other verification equations of $c = 1$ are satisfied if the servers compute the correct values.

Next, the soundness is proved as follows. Assume that the servers correctly answer both $c = 0$ and $c = 1$ cases for the same λ, λ' and $(\tilde{m}_{\ell,1}, \ldots, \tilde{m}_{\ell,N})$, $(\tilde{m'}_{\ell,1}, \ldots, \tilde{m'}_{\ell,N'})$, $\tilde{E}_\ell$ and $\tilde{F}_\ell$. Then, by using φ and λ, one can extract π as $\lambda\varphi^{-1} = \lambda(\pi^{-1}\lambda)^{-1} = \pi$. Similarly, π' is extracted. Furthermore, by using s and w, one can extract t as $s/w = s/(s/t) = t \pmod n$. Though it is complex to extract the knowledge of the individual server, it can be extracted by the similar way to Abe's proof [8].

Finally, to prove the zero-knowledge, ℓ simulators $\mathcal{S}_1, \ldots, \mathcal{S}_\ell$ are constructed as well as Abe's proof [8]. First, the simulators cooperatively choose $c \in_U \{0, 1\}$. If $c = 0$, they honestly conduct the protocol. They can accomplish it, since the knowledge is not needed in this case. If $c = 1$, each simulator $\mathcal{S}_i$ chooses fake permutations $\tilde{\lambda}_i$ and $\tilde{\lambda}'_i$ and a fake factor $\tilde{s}_i \in_U \mathcal{Z}^*_n$. Then, the simulators except the last simulator $\mathcal{S}_\ell$ honestly obey the protocol. In Step 1, the last simulator $\mathcal{S}_\ell$ computes

$$\begin{aligned}\tilde{m}_{\ell,j} &= \dot{m}^{\tilde{s}_\ell}_{\tilde{\lambda}_\ell(j)} \text{ for all } 1 \le j \le k, \\ \tilde{m'}_{\ell,j'} &= \dot{m'}^{\tilde{s}_\ell}_{\tilde{\lambda'}_\ell(j')} \text{ for all } 1 \le j' \le k', \\ \tilde{E}_\ell &= \dot{C}_1^{\tilde{s}_\ell}, \text{and} \\ \tilde{F}_\ell &= \dot{C}_2^{\tilde{s}_\ell},\end{aligned}$$

and sends them to V and all simulators. In Step 3, $\mathcal{S}_\ell$ sends V and all simulators $\varphi = \tilde{\lambda}_\ell, \varphi' = \tilde{\lambda}'_\ell$ and $w = \tilde{s}_\ell$, which satisfy the verification equations of Step 4. The views of the simulators and servers (in the real protocol) are indistinguishable except the ℓ-th party in the case of $c = 1$, since they honestly obey the protocol. Consider S_ℓ and $\mathcal{S}_\ell$ in the case of $c = 1$. In Step 1, $\tilde{m}_{\ell,j}$ of S_ℓ is the form g^R for a random factor $R \in_U \mathcal{Z}^*_n$, since the original m_j is the form and it is raised to the power $s_i \in_U \mathcal{Z}^*_n$. On the other hand, from the same reason, $\dot{m}_j$ is also the form, and so is $\tilde{m}_{\ell,j}$ of $\mathcal{S}_\ell$. Thus, the distributions of $\tilde{m}_{\ell,j}$ of S_ℓ and $\mathcal{S}_\ell$ are the same. It similarly holds for the other values in Step 1. In Step 3, the values w of S_ℓ and $\mathcal{S}_\ell$ distribute uniformly on $\mathcal{Z}^*_n$ and so do the permutations φ, φ'. Therefore, the views of them are also indistinguishable. □

Now, we discuss that the proposed construction satisfies the requirements of the distributed plaintext membership test protocol in Definition 3.

Correctness: Owing the soundness of the shuffle proof part, the correctness of the shuffle part is assured. Thus, for inputs the lists of plaintexts $L = (m_1, \ldots, m_k)$, $L' = (m'_1, \ldots, m'_{k'})$, and an ElGamal ciphertext (C_1, C_2), the outputs $\dot{L} = (\dot{m}_1, \ldots, \dot{m}_k)$, $\dot{L}' = (\dot{m'}_1, \ldots, \dot{m'}_{k'})$ and $(\dot{C}_1, \dot{C}_2)$ satisfy that

$$\begin{aligned}\dot{m}_j &= m^t_{\pi(j)} \text{ for all } 1 \le j \le k, \\ \dot{m'}_{j'} &= m'^t_{\pi'(j')} \text{ for all } 1 \le j' \le k', \\ \dot{C}_1 &= C_1^t, \text{and} \\ \dot{C}_2 &= C_2^t,\end{aligned}$$

for random permutations π, π', and $t \in_U \mathcal{Z}^*_n$. Let m be the plaintext of (C_1, C_2). Then, since $C_1 = g^r$ and $C_2 = y^r m$ hold for a random factor $r \in_U \mathcal{Z}^*_n$, $\dot{C}_1 = g^{rt}$ and $\dot{C}_2 = y^{rt} m^t$ should hold. In the match part, the decrypted $\dot{m}$ is m^t, whose correctness is assured by the proof of the threshold decryption in the match proof part. Therefore, if the plaintext m is in L and/or L', $\dot{m}$ is in $\dot{L}$ and/or $\dot{L}'$, respectively, and vice

versa.

Unlinkability: Owing to the zero-knowledge of the proof parts, the parts reveal no information. Owing to the shuffle part, as Lemma 1 shows, no one can link the plaintexts in L and L' to the randomized plaintexts in $\dot{L}$ and $\dot{L}'$. Furthermore, since the final decrypted $\dot{m}$ is also randomized by t, it cannot be linked to the original plaintext m. Therefore, the unlinkability is satisfied.

5.4 Extension to the Cases with More Ciphertexts

Now, we show the simple extension to the cases with more ciphertexts as inputs: The servers are given N ElGamal ciphertexts $(C_{11}, C_{21}), \ldots, (C_{1N}, C_{2N})$ together with lists L and L'. Then, instead of (C_1, C_2) in the shuffle part, every (C_{1i}, C_{2i}) is randomized by the same t_i as one used for the plaintexts in L and L'. Similarly, in the shuffle proof part, every (C_{1i}, C_{2i}) is randomized by the same s_i instead of (C_1, C_2), and every randomization is verified. In the match and its proof parts, the ciphertexts randomized from $(C_{11}, C_{21}), \ldots, (C_{1N}, C_{2N})$ are all decrypted and the correctness is proved.

The computation cost of this extended construction is $O(N + K)$, where K is the total number of the plaintexts in L and L'. This is better than the cost of the construction by the mix and match protocol. In the other hand, the extended construction allows anyone to check whether the plaintext of a ciphertext (C_{1i}, C_{2i}) is the same as that of another ciphertext (C_{1j}, C_{2j}). However, we consider that this information is little influential in identifying the users in the application to the survey system, as discussed in Section 4.3.

6. Conclusion

In this paper, an anonymous statistical survey system of attributes is proposed, where both verifiability of the correctness and the anonymity are satisfied. Though the complexity of users' offering their attributes is comparable to the practical group signature [6], the complexity of the generating protocol is proportional to the number of all registered users, owing that the distributed plaintext membership test protocol has the complexity that is proportional to the number of all plaintexts. This implies the inefficiency when many users join the system in order to obtain the attributes of many users. Thus, our future work is to propose the system overcoming the inefficiency.

References

1) Nakanishi, T. and Sugiyama, Y.: Anonymous Statistical Survey of Attributes with Correctness, *Proc. Computer Security Symposium 2000* (*CSS2000*), IPSJ Symposium series, Vol.2000, No.12, pp.67–72 (2000).
2) Nakanishi, T. and Sugiyama, Y.: Anonymous Statistical Survey of Attributes, *Proc. 6th Australasian Conference on Information Security and Privacy* (*ACISP2001*), LNCS 2119, pp.460–473, Springer-Verlag (2001).
3) Sako, K.: Generating Statistical Information in Anonymous Surveys, *IEICE Trans. Fundamentals*, Vol.E79-A, No.4, pp.507–512 (1996).
4) Cramer, R., Damgard, I. and Schoenmakers, B.: Proofs of Partial Knowledge and Simplified Design of Witness Hiding Protocols, *Advances in Cryptology — CRYPTO '94*, LNCS 839, pp.174–187, Springer-Verlag (1994).
5) Cramer, R., Franklin, M., Schoenmakers, B. and Yung, M.: Multi-Authority Secret-Ballot Elections with Linear Work, *Advances in Cryptology — EUROCRYPT '96*, LNCS 1070, pp.72–83, Springer-Verlag (1996).
6) 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).
7) Desmedt, Y. and Frankel, Y.: Threshold Cryptosystems, *Advances in Cryptology — CRYPTO'89*, LNCS 435, pp.307–315, Springer-Verlag (1990).
8) Abe, M.: Universally Verifiable Mix-net with Verification Work Independent of the Number of Mix-centers, *Advances in Cryptology — EUROCRYPT'98*, LNCS 1403, pp.437–447, Springer-Verlag (1998).
9) Jakobsson, M. and Juels, A.: Mix and Match: Secure Function Evaluation via Ciphertexts, *Advances in Cryptography — ASIACRYPT2000*, LNCS 1976, pp.162–177, Springer-Verlag (2000).
10) Pedersen, T.P.: Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing, *Advances in Cryptology — CRYPTO '91*, LNCS 1361, pp.129–140, Springer-Verlag (1991).

(Received November 30, 2001)
(Accepted June 4, 2002)

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. Since 2000, he has been a research associate in the Department of Communication Network Engineering, Okayama University, Okayama, Japan. 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.