IEICE TRANS. FUNDAMENTALS, VOL.E88–A, NO.1 JANUARY 2005

PAPER Special Section on Cryptography and Information Security

Efficient and Verifiable Shuffling and Shuffle-Decryption

Jun FURUKAWA[†a)], *Nonmember*

SUMMARY In this paper, we propose an efficient protocol for proving the correctness of shuffling and an efficient protocol for simultaneously proving the correctness of both shuffling and decryption. The former protocol is the most efficient in computational and communication complexity among 3-move honest verifier perfect zero-knowledge protocols for proving a shuffling of ElGamal cipher-texts. The latter protocol is the most efficient in computational, communication, and round complexity, as a whole, in proving the correctness of both shuffling and decryption of ElGamal cipher-texts. The proposed protocols will be a building block of an efficient, universally verifiable mix-net, whose application to voting systems is prominent.

key words: *voting, shuffle, decryption, complete permutation hiding*

1. Introduction

A mix-net [3] protocol is useful in applications which require anonymity, such as voting. Crucial to a mix-net protocol is the execution of multiple rounds of shuffling and decryption by multiple, independent mixers, so that none of the output decryptions can be linked to any of the input encryptions.

To ensure the correctness of output, it is desirable to achieve the property of universal verifiability. Early studies, such as those by Sako and Kilian [22] and Abe [1], required vast amounts of computation to prove and verify the correctness of a shuffling in a mix-net without sacrificing unlinkability. However, recently proposed protocols [7], [8], [12], [16] were sufficiently efficient and practical. The protocols of [7], [8] use the property of permutation matrixes, and the protocols of [12], [16] use the fact that polynomials remain invariant under the permutations of their roots. The protocols of [8], [12], and [16] require the respective computation of $18k$, $12k$, and $42k$ modular exponentiations to prove and verify the correctness of a shuffling of k data. The protocol of [7] requires $19k$ modular exponentiations to prove and verify both shuffling and decryption. Groth's protocol [12] is the most efficient.

A result of these recent works is that proving the correctness of decryption now costs as much as proving the correctness of shuffling. Hence, decreasing the cost of proving decryption has also become important for mix-net. It is now also important in these protocols to decrease round complexity. The round complexity of the interactive protocols is one of their most important complexity measures. Indeed, substantial research efforts have been invested into reducing round complexity in many cryptographic protocols. In particular, no efficient 3-round zero-knowledge argument for shuffling is known. Indeed, the 3-round protocol proposed in [8] is not zero-knowledge.

In this paper, we propose two protocols. The first protocol is a 3-round special honest-verifier perfect zero-knowledge argument for shuffling, which is the most efficient among less-than-7-rounds zero-knowledge arguments for shuffling. The second protocol is a 5-round protocol that proves the correctness of both shuffling and decryption, which is the most efficient in computational, communication, and round complexity, as a whole, in proving the correctness of both shuffling and decryption of ElGamal cipher-texts.

The first protocol is a refinement of the non zero-knowledge protocol in [8] which is not zero-knowledge. We introduced more random elements into the original protocol to make it turn into a zero-knowledge protocol. We also reduced its computational and communication complexity. The second protocol is based on the first protocol. Here, we reduce the cost for decryption by choosing a strategy of simultaneously proving the correctness of shuffling and decryption. The protocol of shuffle-decryption was first presented in [20]. The technique we used here is the same as that introduced in [7]. However, as is mentioned in [7], the protocol of [7] is not zero-knowledge, and this simultaneously proving technique never yields a zero-knowledge protocol.

Although the second protocol is not zero-knowledge, we propose a formal definition for the core requirement of unlinkability in arguments for ElGamal shuffle-decryption and proved that the proposed argument is secure in the sense of this definition. We call this property "complete permutation hiding (CPH)." If an argument for shuffle-decryption is CPH, polynomially-bounded honest verifiers, who may have some partial information about its permutation, will learn nothing new about its permutation from an interaction with a prover in an **overwhelming** number of cases of input even if the arguments with the same private key are repeatedly executed, whereas if the argument is zero-knowledge, verifiers will learn nothing new in **every** case of input.

The definition of security for argument for shuffle-decryption stated in [7] is "No polynomially bounded adversary can compute any partial information of the permutation from the protocol." Unlike our new definition, this defini-

Manuscript received March 11, 2004.
Manuscript revised July 8, 2004.
Final manuscript received August 30, 2004.

[†]The author is with NEC Corporation, Kawasaki-shi, 211-8666 Japan.
a) E-mail: j-furukawa@ay.jp.nec.com

tion does not mention the case where the verifier has already obtained partial information before the protocol begins and where the shuffle-decryptions with the same private key are repeatedly executed. These cases seem to occur quite often.

A simple combination of two zero-knowledge protocols of an argument for shuffling and of an argument for decryption also does not yield a zero-knowledge protocol since the intermediate state cannot be simulated. And our second protocol is not weaker than this simply combined protocol.

We also analyzed efficiency of proposed protocols. Our first protocol requires roughly $15k$ exponentiations, $1344k$ bits of communication, and three rounds to prove in perfect zero-knowledge the correctness of a shuffling of k-data. The most efficient previously known 3-round perfect zero-knowledge proof or argument for shuffling is the protocol of [22], which requires vast amounts of computation, i.e., $640k$ modular exponentiations. Our second protocol requires roughly $14k$ exponentiations, $1344k$ bits of communication, and five rounds to prove and verify the correctness of both shuffling and decryption of k-data. To prove and verify the correctness of both shuffling and decryption of k-data with Groth's protocol [12] by using the standard technique[†] for proving the correctness of decryption, we require $15k$ exponentiations, $2528k$ bits of communication, and seven rounds.

Our paper is organized as follows. Section 2 introduces the properties of permutation matrixes. Section 3 proposes a 3-round honest verifier perfect zero-knowledge argument for shuffling. Section 4 proposes a protocol by which we are able to simultaneously prove the correctness of a shuffle-decryption in an efficient way. Section 5 proposes a definition for the requirement of unlinkability in arguments for shuffle-decryption. Section 6 compares the efficiency of our protocols to prior work.

2. Permutation Matrix

We define a permutation matrix $(A_{ij})_{i,j=1,\cdots,k}$ as follows:

Definition 1: Let q be a prime. A matrix $(A_{ij})_{i,j=1,\cdots,k}$ is a *permutation matrix over* $\mathbb{Z}/q\mathbb{Z}$ if it satisfies

$$A_{ij} = \begin{cases} 1 \bmod q & \text{if } \phi(i) = j \\ 0 \bmod q & \text{otherwise} \end{cases}$$

for a permutation function $\phi : \{1, \ldots, k\} \to \{1, \ldots, k\}$.

The following theorems are the key observations used to construct the proposed protocols.

Theorem 1: ([8] Theorem 1) Let q be a prime. A matrix $(A_{ij})_{i,j=1,\ldots,k}$ is a permutation matrix over $\mathbb{Z}/q\mathbb{Z} \Leftrightarrow$

$$\sum_{h=1}^{k} A_{hi}A_{hj}A_{hg} = \delta_{ijg} \tag{1}$$

$$\sum_{h=1}^{k} A_{hi}A_{hj} = \delta_{ij} \tag{2}$$

for all i, j, and g.

Here

$$\delta_{ij} \triangleq \begin{cases} 1 \pmod q, & \text{if } i = j \\ 0 \pmod q, & \text{if } i \neq j \end{cases}$$

$$\delta_{ijg} \triangleq \delta_{ij}\delta_{jg}$$

Proof. See Theorem 1 in [8] or its copy in Appendix C. □

Theorem 2: For $q \bmod 3 = 2$, a matrix $(A_{ij})_{i,j=1,\ldots,n}$ is a permutation matrix over $\mathbb{Z}/q\mathbb{Z} \Leftrightarrow$ Eq. (1) holds.

Proof. ($\Rightarrow$) is trivial. ($\Leftarrow$); From the proof of Theorem 1 in [8], if Eq.(1) holds, then there is only one non-zero element e_i in each i-th row and it must satisfy $e_i^3 = 1 \bmod q$. Because $q \bmod 3 = 2$ implies that 1 is the only cubic root of 1 in $\mathbb{Z}/q\mathbb{Z}$, e_i must be 1. Therefore, matrix $(A_{ij})_{i,j=1,\ldots,n}$ is a permutation matrix over $\mathbb{Z}/q\mathbb{Z}$. □

The soundness of our first protocol depends directly on Theorem 1. The soundness of our second protocol depends directly on Theorem 2.

3. Perfect Zero-Knowledge Argument for Shuffling

In this section, we propose a 3-round honest verifier perfect zero-knowledge argument for shuffling which is the most efficient among all previous less-than-7-round zero-knowledge arguments for shuffling.

3.1 Notation

Let p, q be two primes s.t. $q|p-1$, $\mathbb{G}_q$ be an order q subgroup of $(\mathbb{Z}/p\mathbb{Z})^*$, $g_0(\neq 1)$ be an element of $\mathbb{G}_q$, $x \in \mathbb{Z}/q\mathbb{Z}$ be a private key, $m_0 = {g_0}^x \bmod p$ be a public key used for re-encryption in shuffling, and k be the number of ElGamal cipher-texts to be shuffled. Let $\{(g_i, m_i) \in \mathbb{G}_q^2\}_{i=1,\ldots,k}$ be ElGamal cipher-texts to be shuffled by the public-key m_0. The resulting cipher-texts is denoted as $\{(g'_i, m'_i) \in \mathbb{G}_q^2\}_{i=1,\ldots,k}$. Treating the public key g_0, m_0 as if it were an element in a cipher-text vector may be awkward, but it gives a more compact and unified representation to variables. Let P be a prover who shuffles and proves the correctness of shuffling to a verifier V.

Let $F_k \triangleq \{f_\nu \in_R \mathbb{G}_q\}_{\nu=-4,\ldots,k}$ be $k+5$ elements of $\mathbb{G}_q$ that are uniformly and randomly generated so that neither P nor V can generate non-trivial integers $a, \{a_\nu\}_{\nu=-4,\ldots,k}$ satisfying $g_0^a \prod_{\nu=-4}^{k} {f_\nu}^{a_\nu} \equiv 1 \pmod p$ with non-negligible probability. The public key is a set, $\{p, q, g_0, m_0, F_k\}$.

3.2 ElGamal Shuffling

ElGamal shuffling is a procedure that, given g_0, m_0 and k ElGamal cipher-texts $\{(g_i, m_i)\}_{i=1,\ldots,k}$, outputs ElGamal cipher-texts

$$(g'_i, m'_i) = ({g_0}^{s_i} g_{\phi^{-1}(i)}, {m_0}^{s_i} m_{\phi^{-1}(i)}) \bmod p \tag{3}$$

for $i = 1, \ldots, k$ where $\{s_i \in_R \mathbb{Z}/q\mathbb{Z}\}_{i=1,\cdots,k}$ and a permutation of indices $\phi : \{1, \ldots, k\} \to \{i = 1, \ldots, k\}$ are chosen uniformly and randomly.

Shuffling of ElGamal cipher-texts results in the follow-

[†]This technique is explained in Section 6.

ing two important properties:

i. There exists a permutation ϕ such that equations $D_x((g'_i, m'_i)) = D_x((g_{\phi^{-1}(i)}, m_{\phi^{-1}(i)}))$ hold for all i. Here, $D_x(\cdot)$ is a decryption algorithm that uses the private key x that corresponds to m_0.
ii. As long as the decision Diffie-Hellman problem is difficult to solve, no polynomially bounded algorithm, given only $p, q, g_0, m_0, \{(g_i, m_i)\}_{i=1,\cdots,k}$, $\{(g'_i, m'_i)\}_{i=1,\cdots,k}$, has an advantage over the random-guessing algorithm in guessing any part of permutation ϕ for uniformly and randomly chosen g_0, m_0, s_i, ϕ.

Using a permutation matrix (A_{ji}), which corresponds to a permutation ϕ, we find that Eq. (3) can be expressed as

$$(g'_i, m'_i) = \left({g_0}^{s_i} \prod_{j=1}^{k} g_j^{A_{ji}}, {m_0}^{s_i} \prod_{j=1}^{k} m_j^{A_{ji}} \right) \bmod p. \tag{4}$$

Therefore, proving the correctness of the shuffle is equivalent to proving the existence of an $s_i \in \mathbb{Z}/q\mathbb{Z}$ for $i = 1, \ldots, k$ and a permutation matrix $(A_{ji})_{i,j=1,\ldots,k}$ which satisfies Eq. (4).

3.3 Protocol Structure

The zero-knowledge argument for shuffling we will propose in this section is almost the same as the protocol proposed in [8]. The proposed protocol and the protocol of [8] are roughly composed of three components. Those in [8] are, (i) generation of $\{f'_i\}_{i=1,\cdots,k}$ and an argument of knowledge of s_i and a general matrix (A_{ji}) that satisfy

$$f'_i = f_0^{s_i} \prod_{j=1}^{k} f_j^{A_{ji}} \bmod p \quad i = 1, \cdots, k, \tag{5}$$

for uniformly and randomly chosen $\{f_\mu \in_R \mathbb{G}_q\}_{\mu=0,\ldots,k}$, (ii) proof that (A_{ji}) whose knowledge proved in (i) is a permutation matrix using Theorem 1, and (iii) proof that s_i and (A_{ji}) whose knowledge is proved in (i) also satisfies Eq. (4). In Proof (ii), there are commitment, challenge, and response phase.

The main difference between our protocol and the protocol of [8] is that we have introduced the values $f_{-4}, f_{-3}, f_{-2}, f_{-1}$ in the proposed protocol. Because of these values f'_is in the commitment are modified from $f'_i = f_0^{A_{0i}} f_{\phi^{-1}(i)}$ to $f'_i = f_{-4}^{A_{-4i}} f_{-3}^{A_{-3i}} f_{-2}^{A_{-2i}} f_{-1}^{A_{-1i}} f_0^{A_{0i}} f_{\phi^{-1}(i)}$. As a result, we have more redundancy $(A_{-4i}, A_{-3i}, A_{-2i}, A_{-1i})$ to generate the value $\{f'_i\}_{i=1,\ldots,k}$. Then we adjusted $A_{-4i}, A_{-3i}, A_{-2i}$ so that some values in the commitment become zero, which decreased the number of terms in checking equations in the response phase†. And because of the redundancy gained by A_{-1i}, the proposed protocol enjoys zero-knowledgeness unlike the protocol in [8].

The other difference between them is with respect to the verification of Eq. (9). A verification that Eq. (9) holds is equivalent to verifying that equations

$$\prod_{\nu=-4}^{k} f_\nu^{r_\nu} = f'_0 \prod_{i=1}^{k} f_i'^{c_i} \pmod p$$

$$\prod_{\nu=-4}^{k} f_\nu^{r'_\nu} = \tilde{f}'_0 \prod_{i=1}^{k} f_i'^{{c_i}^2} \pmod p$$

hold. The verifier is able to additionally check that the second equation holds with negligible overhead. Since, r'_{-2} in this second equation plays the role of λ' in [8], we can omit Equation (14) in [8].

3.4 Proposed Zero-Knowledge Argument

Let us next introduce our argument for shuffling.

3.4.1 ElGamal Shuffling

P uniformly and randomly chooses $\{A_{0i} \in_R \mathbb{Z}/q\mathbb{Z}\}_{i=1,\cdots,k}$ and a permutation matrix $(A_{ji})_{i,j=1,\cdots,k}$. P shuffles k ElGamal ciphertexts $\{(g_i, m_i)\}_{i=1,\cdots,k}$ to $\{(g'_i, m'_i)\}_{i=1,\cdots,k}$ as

$$\begin{aligned}(g'_i, m'_i) &= \left(\prod_{\nu=0}^{k} {g_\nu}^{A_{\nu i}}, \prod_{\nu=0}^{k} {m_\nu}^{A_{\nu i}} \right) \bmod p \\ &= ({g_0}^{A_{0i}} g_{\phi^{-1}(i)}, {m_0}^{A_{0i}} m_{\phi^{-1}(i)}) \bmod p.\end{aligned} \tag{6}$$

Here, the matrix $(A_{\nu i})_{\nu=0,\ldots,k;i=1,\ldots,k}$ in Eq. (6) is a $(k+1) \times k$ matrix that contains a permutation matrix whose size is $k \times k$ as a sub-matrix. It should be remembered here that, in the rest of the paper, just as the size of matrices may often change, so will the range over which the indices of the matrix A run.

3.4.2 Proving a Shuffle

Commitment: P uniformly and randomly chooses $\{A_{\nu 0}, A'_\nu\}_{\nu=-4,\ldots,k}$ and $\{A_{-1i}\}_{i=1,\cdots,k}$ from $\mathbb{Z}/q\mathbb{Z}$, and then computes:

$$A_{-2i} = \sum_{j=1}^{k} 3{A_{j0}}^2 A_{ji} \bmod q \quad i = 1, \cdots, k$$

$$A_{-3i} = \sum_{j=1}^{k} 3A_{j0}A_{ji} \bmod q \quad i = 1, \cdots, k$$

$$A_{-4i} = \sum_{j=1}^{k} 2A_{j0}A_{ji} \bmod q \quad i = 1, \cdots, k$$

$$f'_\mu = \prod_{\nu=-4}^{k} f_\nu^{A_{\nu\mu}} \bmod p \quad \mu = 0, \cdots, k \tag{7}$$

$$\tilde{f}'_0 = \prod_{\nu=-4}^{k} f_\nu^{A'_\nu} \bmod p \tag{8}$$

$$g'_0 = \prod_{\nu=0}^{k} {g_\nu}^{A_{\nu 0}} \bmod p$$

†The equations in [8] that correspond to Equations (12) and (13) are Equations (15) and (16) in [8]. We can see that terms quadratic and linear to elements of the challenge disappear in the proposed protocol.

$$m'_0 = \prod_{\nu=0}^{k} m_\nu{}^{A_{\nu 0}} \bmod p$$

$$w = \sum_{j=1}^{k} A_{j0}{}^3 - A_{-20} - A'_{-3} \bmod q$$

$$\dot{w} = \sum_{j=1}^{k} A_{j0}{}^2 - A_{-40} \bmod q.$$

Then, P sends $g'_0, m'_0, \tilde{f}'_0, \{f'_\mu\}_{\mu=0,\cdots,k}, w, \dot{w}$ to V as a commitment.

Challenge: V uniformly and randomly chooses $\{c_i \in \mathbb{Z}/q\mathbb{Z}\}_{i=1,\cdots,k}$ and sends it to P.

Response: P sends V the following response:

$$r_\nu = \sum_{\mu=0}^{k} A_{\nu\mu} c_\mu \bmod q \quad \nu = -4, \cdots, k$$

$$r'_\nu = \sum_{i=1}^{k} A_{\nu i} c_i{}^2 + A'_\nu \quad \nu = -4, \cdots, k$$

where $c_0 = 1 \bmod q$.

Verification:

V accepts the shuffle if the following equations hold for a uniformly and randomly chosen $\alpha \in_R \mathbb{Z}/q\mathbb{Z}$:

$$\prod_{\nu=-4}^{k} f_\nu{}^{r_\nu + \alpha r'_\nu} \equiv f'_0 \tilde{f}'_0{}^{\alpha} \prod_{i=1}^{k} f'_i{}^{c_i + \alpha c_i{}^2} \pmod p \tag{9}$$

$$\prod_{\nu=0}^{k} g_\nu{}^{r_\nu} \equiv \prod_{\mu=0}^{k} g'_\mu{}^{c_\mu} \pmod p \tag{10}$$

$$\prod_{\nu=0}^{k} m_\nu{}^{r_\nu} \equiv \prod_{\mu=0}^{k} m'_\mu{}^{c_\mu} \pmod p \tag{11}$$

$$\sum_{j=1}^{k} (r_j^3 - c_j^3) \equiv r_{-2} + r'_{-3} + w \pmod q \tag{12}$$

$$\sum_{j=1}^{k} (r_j^2 - c_j^2) \equiv r_{-4} + \dot{w} \pmod q. \tag{13}$$

View: The view of this protocol is

$$p, q, g_0, m_0, \{(g_i, m_i)\}_{i=1,\ldots,k}, \{(g'_i, m'_i)\}_{i=1,\ldots,k},$$
$$\{f_\nu\}_{\nu=-4,\ldots,k}, \{f'_\mu\}_{\mu=0,\ldots,k}, \tilde{f}'_0, g'_0, m'_0,$$
$$w, \dot{w}, \{c_i\}_{i=1,\ldots,k}, \{r_\nu\}_{\nu=-4,\ldots,k}, \{r'_\nu\}_{\nu=-4,\ldots,k}$$

3.5 Properties of the Proposed Argument for Shuffling

Theorem 3: The protocol is complete.

Proof. It is clear. □

Theorem 4: The protocol is sound as long as the discrete logarithm problem is difficult to solve.

Proof. The proof is given in Appendix A.1 □

Theorem 5: The protocol is honest-verifier perfect zero-knowledge.

Proof. The proof is given in Appendix A.2 □

4. An Efficient Argument for Shuffle-Decryption

In this section, we propose an argument for shuffle-decryption that is the most efficient of all previous protocols, as a whole, to prove the correctness of both shuffling and decryption of ElGamal cipher-texts. The protocol requires five rounds. Although the protocol is not zero-knowledge, we can prove that the verifier learn nothing new about its permutation.

4.1 Notation

Let p, q be two primes such that $q|p-1$ and $q \bmod 3 = 2$. Note that we have imposed an additional condition to q. Let $\mathbb{G}_q$ be an order q subgroup of $(\mathbb{Z}/p\mathbb{Z})^*$, $g_0 (\neq 1)$ be an element of $\mathbb{G}_q$, k be the number of ElGamal cipher-texts to be shuffled, and ℓ be the number of shuffler-decrypters. Let $x'^{(\lambda)} \in_R \mathbb{Z}/q\mathbb{Z}$ be a private key of the λ-th shuffle-decrypter which is used for the partial decryption and $y^{(\lambda)} = g_0{}^{x'^{(\lambda)}} \bmod p$ be the corresponding public key. Let $m_0{}^{(\lambda)} = \prod_{\kappa=1}^{\lambda} y^{(\kappa)} \bmod p$ be a public key used for the shuffling of the λ-th shuffle-decrypter.

Let $\{(g_i^{(\ell)}, m_i^{(\ell)})\} = \{(g_0{}^{\bar{r}_i}, (m_0^{(\ell)})^{\bar{r}_i} M_i)\}_{i=1,\ldots,k} \bmod p$ be ElGamal cipher-texts to be input to the ℓ-th shuffle-decrypter where $\{M_i \in \mathbb{G}_q\}_{i=1,\ldots,k}$ is a set of encrypted plain texts, and $\{\bar{r}_i \in_R \mathbb{Z}/q\mathbb{Z}\}_{i=1,\ldots,k}$ be elements of $\mathbb{Z}/q\mathbb{Z}$, which are uniformly and randomly chosen. Let $\{(g_i^{(\lambda)}, m_i^{(\lambda)})\}_{i=1,\ldots,k}$ be cipher-texts to be input to the λ-th shuffle-decrypter, who shuffles them with the public key $(g_0, m_0^{(\lambda)})$ and then partially decrypts them with the private key $x'^{(\lambda)}$. The resulting cipher-texts are $\{(g_i'^{(\lambda)}, m_i'^{(\lambda)})\}_{i=1,\ldots,k} = \{(g_i^{(\lambda-1)}, m_i^{(\lambda-1)})\}_{i=1,\ldots,k}$ which are passed to the $(\lambda-1)$-th shuffle-decrypter. In the rest of the paper, we only consider the λ-th shuffle-decrypter and omit the index (λ).

We assume another public key $F_k \triangleq \{f_\nu \in_R \mathbb{G}_q\}_{\nu=-2,\ldots,k}$ which are $k+3$ elements of $\mathbb{G}_q$ that are uniformly and randomly generated so that neither P nor V can generate non-trivial integers $a, \{a_\nu\}_{\nu=-2,\ldots,k}$ satisfying $g_0^a \prod_{\nu=-2}^{k} f_\nu{}^{a_\nu} \equiv 1 \pmod p$ with non-negligible probability. The public key is a set, $\{p, q, g_0, m_0, y, F_k\}$.

4.2 ElGamal Shuffle-Decryption

ElGamal shuffle-decryption is a combination procedure of ElGamal shuffling and partial decryption that, given g_0, m_0, y and k ElGamal cipher-texts $\{(g_i, m_i)\}_{i=1,\cdots,k}$, outputs ElGamal cipher-texts

$$(g'_i, m'_i) = (g_0{}^{s_i} g_{\phi^{-1}(i)}, g'^{-x'}_i m_0{}^{s_i} m_{\phi^{-1}(i)}) \bmod p$$

for $i = 1, \ldots, k$ where $\{s_i \in_R \mathbb{Z}/q\mathbb{Z}\}_{i=1,\cdots,k}$ and ϕ are chosen uniformly and randomly. Here, the multiplication by $g'^{-x'}_i$ in the second term has the effect of partial decryption[†].

4.3 Protocol Structure and Tricks for Efficiency

In [7], a technique to simultaneously prove the correctness of shuffling and decryption is proposed. The verifiable shuffle-decryption protocol proposed in [7] is the result of application of this technique to the protocol proposed in [8]. The argument for shuffle-decryption we will propose in this section is the result of application of this simultaneously proving technique to the zero-knowledge argument for shuffling proposed in the previous section with little modification. We briefly sketch this technique.

We first consider the following 4-round zero-knowledge proof for decryptions of cipher-texts. The prover wants to prove that $\{(g'_i, m'_i)\}_{i=1,\ldots,k}$ is the valid partial decryptions of $\{(g'_i, \bar{m}_i)\}_{i=1,\ldots,k}$ with a private key x' such that $y = {g_0}^{x'} \bmod p$, that is, the relation $m'_i/\bar{m}_i = {g'_i}^{x} \bmod p$ holds for all $i = 1, \ldots, k$.

i. The verifier sends to the prover uniformly and randomly chosen $\{c_j \in_R \mathbb{Z}/q\mathbb{Z}\}_{j=1,\ldots,k}$.
ii. The prover chooses $z' \in_R \mathbb{Z}/q\mathbb{Z}$ uniformly and randomly and then computes

$$\zeta = \prod_{j=1}^{k} {g'_j}^{c_j} \bmod p$$
$$y' = g^{z'}\ ,\ \eta' = \zeta^{z'} \bmod p$$

and sends y', η' to the verifier.
iii. The verifier sends to the prover uniformly and randomly chosen $c' \in_R \mathbb{Z}/q\mathbb{Z}$.
iv. The prover sends to the verifier

$$r' = c'x' + z' \bmod q$$

v. The verifier computes

$$\zeta = \prod_{j=1}^{k} {g'_j}^{c_j}\ ,\ \eta = \prod_{j=1}^{k} (m'_j/\bar{m}_j)^{c_j} \bmod p$$

and accepts if

$$g^{r'} = y^{c'}y'\ ,\ \zeta^{r'} = \eta^{c'}\eta' \bmod p$$

hold.

We have noticed that some of the computations done in the proposed argument for shuffling in the previous section and in this proof for decryption are similar. For example, $\zeta = \prod_{j=1}^{k} {g'_j}^{c_j} \bmod p$ with respect to the verifier's challenge $\{c_j\}$, is also computed in the argument for shuffling, $\eta = \prod_{j=1}^{n} (m'_j/\bar{m}_j)^{c_j} = \prod_{j=1}^{k} {m'_j}^{c_j} / \prod_{j=1}^{k} {\bar{m}_j}^{c_j} \bmod p$ in the proof for decryption and $\prod_{j=1}^{k} {\bar{m}_j}^{c_j} \bmod p$ in the argument for shuffling have the same factor.

We merged the two protocols in the following way: Two proofs share the same challenge $\{c_i\}_{i=1,\ldots,k}$ so that both prover and verifier need to compute the above mentioned values only once. Since η, which is a combination of $\{g'_j\}_{i=1,\ldots,k}$ and the challenge $\{c_i\}_{i-1,\ldots,k}$, does not appear in the protocol proposed in [12], we cannot apply this technique directly to that protocol.

Although, as is mentioned in [7], the protocol resulting from this simultaneously proving technique cannot be zero-knowledge, we can prove its security. Moreover, since we do not require the protocol to be zero-knowledge any more, we can modify the protocol, so long as the protocol remains secure at the same security level, so as to be more efficient. In particular, the protocol requires less random numbers to remain at the same security level, thus we reduce the set of extra public parameters $\{f_{-4}, f_{-3}, f_{-2}, f_{-1}\}$ to $\{f_{-3}, f_{-2}, f_{-1}\}$.

We have made one more modification to the original protocol. Since the proposed protocol adopts the prime q such that $q \bmod 3 = 2$, verifiers do not need to confirm that Equation (2) holds[††] any more. Hence we do not need r_{-4} and reduce the set of extra public parameters $\{f_{-3}, f_{-2}, f_{-1}\}$ to $\{f_{-2}, f_{-1}\}$.

4.4 Proposed Argument

We now describe our proposed argument for ElGamal shuffle-decryption.

4.4.1 ElGamal Shuffle-Decryption

P uniformly and randomly chooses $\{A_{0i} \in_R \mathbb{Z}/q\mathbb{Z}\}_{i=1,\cdots,k}$ and a permutation matrix $(A_{ji})_{i,j=1,\cdots,k}$ and then shuffles and decrypts k ElGamal cipher-texts $\{(g_i, m_i)\}_{i=1,\cdots,k}$ to $\{(g'_i, m'_i)\}_{i=1,\cdots,k}$ as

$$\begin{aligned}(g'_i, m'_i) &= ({g_0}^{A_{0i}} g_{\phi^{-1}(i)}, {g'_i}^{-x'} {m_0}^{A_{0i}} m_{\phi^{-1}(i)}) \bmod p \\ &= \left(\prod_{\nu=0}^{k} {g_\nu}^{A_{\nu i}}, {g'_i}^{-x'} \prod_{\nu=0}^{k} {m_\nu}^{A_{\nu i}}\right) \bmod p. \end{aligned} \tag{14}$$

In this protocol, the witness W_n is a set $\{x', (A_{ji})_{i,j=1,\ldots,k}, \{A_{0i}\}_{i=1,\ldots,k}\}$, and the common input X_n is a set $\{p, q, y, g_0, m_0, F_k, (g_i, m_i)_{i=1,\ldots,k}, (g'_i, m'_i)_{i=1,\ldots,k}\}$. P is given X_n and W_n, and V is given X_n.

4.4.2 Proving a Shuffle-Decryption

Commitment-1: P uniformly and randomly chooses $\{A_{\nu 0}, A'_\nu \in_R \mathbb{Z}/q\mathbb{Z}\}_{\nu=-2,\ldots,k}$ and then computes:

$$A_{-1i} = \sum_{j=1}^{k} 3A_{j0}A_{ji} \bmod q \quad i = 1, \cdots, k$$
$$A_{-2i} = \sum_{j=1}^{k} 3{A_{j0}}^2 A_{ji} \bmod q \quad i = 1, \cdots, k$$

[†]Note that this multiplier is not ${g_i}^{-x'}$ which is part of input cipher-texts. The first element of the left side of the equation is used in the second element of the right side of the equation to define the second element of the left side of the equation.

[††]We can omit a verification of the equation that corresponds to Equation (13).

$$f'_\mu = \prod_{\nu=-2}^{k} f_\nu{}^{A_{\nu\mu}} \bmod p \quad \mu = 0, \cdots, k$$

$$\tilde{f}'_0 = \prod_{\nu=-2}^{k} f_\nu{}^{A'_\nu} \bmod p$$

$$g'_0 = \prod_{\nu=0}^{k} g_\nu{}^{A_{\nu 0}} \bmod p$$

$$m'_0 = \prod_{\nu=0}^{k} m_\nu{}^{A_{\nu 0}} \bmod p$$

$$w = \sum_{j=1}^{k} A_{j0}{}^3 - A_{-20} - A'_{-1} \bmod q$$

Then, P sends $g'_0, m'_0, w, \tilde{f}'_0, \{f'_\mu\}_{\mu=0,\cdots,k}$to V as a commitment.
Challenge-1: V uniformly and randomly chooses $\{c_i\}_{i=1,\cdots,k}$ from $\mathbb{Z}/q\mathbb{Z}$ and sends it to P.
Response-1: P sends V the following response:

$$r_\nu = \sum_{\mu=0}^{k} A_{\nu\mu} c_\mu \bmod q \quad \nu = -2, \cdots, k$$

$$r'_\nu = \sum_{i=1}^{k} A_{\nu i} c_i{}^2 + A'_\nu \bmod q \quad \nu = -2, \cdots, k$$

where $c_0 = 1 \bmod p$.
Commitment-2: P then computes

$$\zeta = \prod_{i=1}^{k} g'_i{}^{c_i} \bmod p.$$

P uniformly and randomly chooses $\beta \in_R \mathbb{Z}/q\mathbb{Z}$, computes the following commitment, and sends it to V:

$$\eta = \zeta^{x'} \bmod p \ , \quad \eta' = \zeta^\beta \bmod p$$
$$y' = g_0{}^\beta \bmod p. \tag{15}$$

Challenge-2: V uniformly and randomly chooses c' from $\mathbb{Z}/q\mathbb{Z}$ and sends it to P.
Response-2: P sends V the following response: $r' = c'x' + \beta \bmod q$
Verification: V computes

$$\zeta = \prod_{i=1}^{k} g'_i{}^{c_i} \bmod p.$$

V accepts the shuffle-decryption if the following equations hold for a uniformly and randomly generated $\alpha \in_R \mathbb{Z}/q\mathbb{Z}$:

$$\prod_{\nu=-2}^{k} f_\nu{}^{r_\nu + \alpha r'_\nu} \equiv f'_0 \tilde{f}'_0{}^\alpha \prod_{i=1}^{k} f'_i{}^{c_i + \alpha c_i{}^2} \pmod p \tag{16}$$

$$\prod_{\nu=0}^{k} g_\nu{}^{r_\nu} \equiv \zeta g'_0 \pmod p \tag{17}$$

$$\prod_{\nu=0}^{k} m_\nu{}^{r_\nu} \equiv \eta \prod_{\mu=0}^{k} m'_\mu{}^{c_\mu} \pmod p \tag{18}$$

$$\sum_{j=1}^{k} (r_j^3 - c_j^3) \equiv r_{-2} + r'_{-1} + w \pmod q \tag{19}$$

$$g_0{}^{r'} \equiv y^{c'} y' \pmod p \tag{20}$$

$$\zeta^{r'} \equiv \eta^{c'} \eta' \pmod p \tag{21}$$

The view $View_V^P(X_n, W_n)$ of this protocol is

$$p, q, y, g_0, m_0, \{(g_i, m_i)\}_{i=1,\ldots,k}, \{(g'_i, m'_i)\}_{i=1,\ldots,k},$$
$$\{f_\nu\}_{\nu=-2,\ldots,k}, f'_0, \{f'_i\}_{i=1,\ldots,k},$$
$$\tilde{f}'_0, g'_0, m'_0, w, \{c_i\}_{i=1,\ldots,k},$$
$$\{r_\nu\}_{\nu=-2,\ldots,k}, \{r'_\nu\}_{\nu=-2,\ldots,k}, \eta, \eta', y', c', r'.$$

4.5 Properties of the Proposed Argument for Shuffle-Decryption

Theorem 6: The protocol is complete.

Proof. It is clear. □

Theorem 7: The protocol is sound as long as the discrete logarithm problem is difficult to solve.

Proof. Proof is given in Appendix B.1. □

If the decision Diffie-Hellman problem is difficult to solve, honest verifiers learn nothing new about its permutation from an interaction with a prover in an overwhelming number of cases of input, even if the protocols with the same private key are repeatedly executed. We formalize this security notion and prove that the protocol satisfies this notion in the next section.

5. Complete Permutation Hiding

5.1 Intuitive Security

The fact that the proposed argument is not less secure than the simple combination of shuffling and decryption is intuitively understood as in the following. Consider the proof system in which a common input to a prover is $p, q, y, g_0, m_0, f_0, \{(g_i, m_i, f_i)\}_{i=1,\ldots,k}, \{(g'_i, \bar{m}_i, \bar{f}'_i)\}_{i=1,\ldots,k}$ and $\{m'_i\}_{i=1,\ldots,k}$. Where

$$(g'_i, \bar{m}_i, \bar{f}'_i) = (g_0{}^{r_i} g_{\pi(i)}, m_0{}^{r_i} m_{\pi(i)}, f_0{}^{r_i} f_{\pi(i)}) \bmod p$$
$$m'_i = g'_i{}^{-x} \bar{m}_i \bmod p.$$

Here, the first equation can be taken as a shuffle of generalized ElGamal cipher-texts composed of three elements instead of two elements. This shuffling as well as the ordinary shuffling, leaks no knowledge about its permutation as long as the decision Diffie-Hellman problem is difficult to solve. The second equation is a partial decryption of shuffled input cipher-texts. The only obstacle that prevents simulating the whole proposed argument for shuffle-decryption is the intermediate state $\{(\bar{m}_i, \bar{f}'_i)\}_{i=1,\ldots,k}{}^\dagger$. This is unsimulatable without the knowledge of $\{r_i\}$ and ϕ. Since the simple

combined protocol also needs to reveal such an unsimulatable intermediate state, we can intuitively conclude that the proposed protocol is no less secure than the simple combination of two perfect zero-knowledge protocol. Since the above shuffled-cipher-texts-like intermediate states are part of the argument, we are required to consider the security of the argument in the same manner in which we consider the security of plain shuffle-decryption.

5.2 Adaptive Chosen Cipher-Text Attack

As discussed in the previous subsection, the proposed argument for ElGamal shuffle-decryption is as secure as the simple combination of shuffling and decryption. However, the security of ElGamal shuffle-decryption itself is not so trivial. This is because, in the mix-net protocol composed of a sequence of shuffle-decryptions, the last shuffle-decrypter outputs a permuted list of "plain-texts." This means mix-net behaves as a kind of decryption oracle. Thus, adaptive chosen cipher-text attack is effective for adversaries.

An example of such an attack can be seen as in the following. Consider the case where a user U_1 sends a plain ElGamal cipher-text to a mix-net composed of ElGamal shuffle-decryptions, and suppose that the mix-net accepts this cipher-text. Then, an adversary may obtain this cipher-text, re-encrypt it, and send the resulting cipher-text to the mix-net. At the end, mix-net outputs a list of plain-texts. If the adversary finds two same plain-texts, then the possibility that this cipher-text is the one that U_1 generated is meaningfully high.

The above observation implies that the security of arguments for shuffle-decryption depends on the security of input cipher-texts. Hence, we assume that cipher-texts input to the mix-net are ElGamal cipher-texts that are secure under adaptive chosen cipher-text attack (IND-CCA2 secure) instead of plain ElGamal cipher-texts. And validity of cipher-text needs to be publicly checkable. An example of such a crypto-system can be found in the literature [25]. This crypto-system can be proved to be IND-CCA2 secure assuming the random oracle model. One can, in this crypto-system, extract the random number used for generating a ElGamal cipher-text by rewinding the user at least with the number of the step quadratic to that of real protocol. We assume input to the mix-net is cipher-texts of such a crypto-system.

5.3 Complete Permutation Hiding

We propose here the notion of *complete permutation hiding* (CPH) as a core requirement of unlinkability in arguments for ElGamal shuffle-decryption. If an argument for shuffle-decryption is CPH, polynomially-bounded honest verifiers, who may have some partial information about its permutation, will learn nothing new about its permutation from an interaction with a prover in an **overwhelming** number of cases of input even if the protocols with the same private key are repeatedly executed, whereas if the protocol is zero-knowledge, verifiers will learn nothing new in **every** case of input.

A verifier in CPH may obtain knowledge about permutation as long as it happens only in a negligible number of cases of common input X_n and witness W_n that the generator G_R (defined below) outputs. This is because we take the success probability of adversaries over the distribution of outputs of G_R.

Let I_n be a *set* of domain parameters $1^n, p, q$, where p and q are primes and are the lengths of the polynomial of n, private key $\bar{x}$, plain texts $\{M_i \in \mathbb{G}_q\}_{i=1,\ldots,k}$, and random tapes of U and G_R. Let $enc(U)$ be an *encoding of a probabilistic polynomial time (PPT) Turing machine U* which generates cipher-texts $\{(g_i, m_i)\}_{i=1,\ldots,k}$ input to the shuffle-decryption procedure. We assume the existence of a knowledge extractor that can extract $\{\bar{r}_i\}_{i=1,\ldots,k}$ such that ${g_0}^{\bar{r}_i} = g_i$ from U.

Existence of such a knowledge extractor can be justified as in the following. Suppose that input cipher-texts $\{(g_i, m_i)\}_{i=1,\ldots,k}$ to shuffle-decrypter are $\{(g_i^{(\lambda)}, m_i^{(\lambda)})\}_{i=1,\ldots,k}$, and suppose that the input cipher-texts to the mix-net are $\{(g_i^{(\ell)}, m_i^{(\ell)})\}_{i=1,\ldots,k}$. Then $\{(g_i^{(\lambda)}, m_i^{(\lambda)})\}_{i=1,\ldots,k}$ are already shuffled and decrypted by $\ell - \lambda + 1$ shuffle-decrypters. If proofs for these shuffles and decryptions are accepted, shuffle-decrypters are able to collaborately generate $\{\tilde{s}_i\}_{i=1,\ldots,k}$ and ϕ such that $g_i^{(\lambda)} = g_{\phi^{-1}(i)}^{(\ell)} {g_0}^{\tilde{s}_i}$. And one can extract $\tilde{s}_i$ such that $g_i^{(\ell)} = {g_0}^{\tilde{s}_i}$ by rewinding the user who generated g_i. Hence, there exists an extractor that can extract $\bar{r}_i$ such that $g_i^{(\lambda)} = {g_0}^{\bar{r}_i}$.

Definition 2: Given $I_n(= \{1^n, p, q, \bar{x} \in \mathbb{Z}/q\mathbb{Z}, \{M_i \in \mathbb{G}_q\}_{i=1,\cdots,k}, Z_n\})$ and $enc(U)$, *instance generator* G_R chooses $g_0 \in_R \mathbb{G}_q, x' \in_R \mathbb{Z}/q\mathbb{Z}$,
$\{s_i \in_R \mathbb{Z}/q\mathbb{Z}\}_{i=1,\ldots,k}$, and a permutation ϕ uniformly and randomly and computes:

$$m_0 = {g_0}^{x'+\bar{x}}, y = {g_0}^{x'} \bmod p$$
$$(g_i, m_i) = U(I_n, g_0, y) \in \mathbb{G}_q \times \mathbb{G}_q$$
$$(g_i', m_i') = (g_0{}^{s_i} g_{\phi^{-1}(i)}, g_i{}^{-x'} m_0{}^{s_i} m_{\phi^{-1}(i)}) \bmod p.$$

G_R then outputs common input X_n and witness W_n:

$$X_n = \{p, q, y, \bar{x}, g_0, m_0, \{(g_i, m_i)\}_{i=1,\cdots,k}, \{(g_i', m_i')\}_{i=1,\cdots,k}\},$$
$$W_n = \{\phi, \{s_i\}_{i=1,\cdots,k}, x'\}.$$

Where, U is a PPT Turing machine such that there exists a knowledge extractor that can extract $\bar{r}_i$ that satisfies $g_i = {g_0}^{\bar{r}_i}$ by rewinding U and choosing the random oracle. And $\bar{x}$ is the sum of other prover's private keys in the mix-net.

In the above definition, U is a PPT Turing machine that plays the role of (malicious and colluding) players who generate cipher-texts $\{(g_i, m_i)\}$. Although U is determined before the public parameter is generated, it does not lose generality because it has this public parameter as an input. In a

[†]Since there is a subtle difference between $\bar{f}_i'$ and f_i', rigorous logic is more complicated.

case where U plays the role of honest players, it outputs

$$(g_i, m_i) = ({g_0}^{\bar{r}_i}, M_i {m_0}^{\bar{r}_i}) \bmod p \quad i = 1, \cdots, k$$

using random numbers $\{\bar{r}_i\}_{i=1,\cdots,k}$.

We say X_n and W_n *satisfy relation* R if the following equations are satisfied:

$$m_0 \equiv {g_0}^{x'+\bar{x}}, y \equiv {g_0}^{x'} \pmod p$$
$$(g'_i, m'_i) \equiv ({g_0}^{s_i} g_{\phi^{-1}(i)}, {g_i}^{-x'} {m_0}^{s_i} m_{\phi^{-1}(i)}) \pmod p.$$

We denote this fact as $(X_n, W_n) \in R$. If there exists a witness W_n for a common input X_n that satisfies $(X_n, W_n) \in R$, common input X_n is a *correct shuffle-decryption*. Generator G_R outputs such an X_n.

Definition 3: Let $View_V^P(X_n, W_n)$ be V's *view of an interaction* with P, which is composed of the common input X_n, messages V receives from P, random tape input to V, and messages V sends to P during joint computation employing X_n, where P has auxiliary input W_n such that $(X_n, W_n) \in R$. $View_V^P$ is an abbreviation of $View_V^P(X_n, W_n)$.

We consider the case when a semi-honest verifier may collude with malicious players who encrypt the cipher-texts and other provers who shuffle and decrypt in the same mix-net. Such a verifier and players may obtain partial information regarding the plain texts $\{M_i\}$, private key $\bar{x}$, random tapes of players, and even a part of the permutation ϕ in addition to $View_V^P$. Moreover, they may obtain the results of other shuffle-decryptions executed by the same prover.

Then it is reasonable to describe this extra information that the verifier may have as $H(I_n, enc(U), X_n, \phi)$ and input cipher-texts generated by the malicious player as $U(I_n, g_0, y)$ using PPT Turing machines $H(\cdot)$ and $U(\cdot)$. Note that $\{s_i\}_{i=1,\cdots,k}$ are not included in the arguments of H, because we consider only the case where the prover never reveals these values to anyone and never uses the same $\{s_i\}_{i=1,\cdots,k}$ for other shuffle-decryptions.

Even though the verifier and the players may obtain the results of other shuffle-decryptions executed by the same prover who uses x', we do not include x' but include y into the input of U and H. Moreover, we assume that there exists a PPT Turing machine K such that the distribution of $View_V^P$ for such H and U and that of $K(I_n, g_0, y, enc(U), \phi)$ are the same. We denote this as $View_V^P \approx K(I_n, g_0, y, enc(U), \phi)$. The exclusion of x' is crucial because it enables us to consider the security of arguments for shuffle-decryption over the distribution of X_n, which includes that of x'. This is exactly the same way that we consider the security of plain shuffle-decryption. We describe information about the permutation ϕ that verifiers try to learn as $f(\phi)$ using PPT Turing machine f. This description can be justified because the expression $f(\phi)$ is sufficient to express any bit of ϕ and any kind of check sum for ϕ.

Now we can say that an argument for shuffle-decryption hides its permutation completely with respect to G_R - i.e., CPH occurs - if there exists a probabilistic polynomial time algorithm ${E'}^E$ (which has black box access to E) with inputs X_n and $H(I_n, enc(U), X_n, \phi)$ that suffers no disadvantage with respect to learning anything about the permutations compared to any probabilistic polynomial time verifier E having input $View_V^P$ and $H(I_n, enc(U), X_n, \phi)$ even if the protocols with the same private key are repeatedly executed. This leads to:

Definition 4: *(complete permutation hiding)* Let $enc(U)$ be an *encoding of a probabilistic polynomial time (PPT) Turing machine* U which generates cipher-texts $\{(g_i, m_i)\}_{i=1,\cdots,k}$ input to the shuffle-decryption procedure. Suppose that there exists a knowledge extractor that can extract $\{\bar{r}_i\}_{i=1,\cdots,k}$ such that ${g_0}^{\bar{r}_i} = g_i$ from U.

Then, an argument for shuffle-decryption (P, V, G_R) achieves *complete permutation hiding* if

$$\begin{aligned} &\exists_{E'^E} \forall_E \forall_H \forall_f \forall_U \forall_c > 0 \exists_N \forall_n > N \forall_{I_n} \\ &\Pr[E(View_V^P, H(I_n, enc(U), X_n, \phi)) = f(\phi)] \\ &< \Pr[{E'}^E(X_n, H(I_n, enc(U), X_n, \phi)) = f(\phi)] \\ &+ \frac{1}{n^c}, \\ &\text{and} \\ &\exists_K \quad View_V^P \approx K(I_n, g_0, y, enc(U), \phi) \end{aligned} \tag{22}$$

where E', E, H, f, U, K are PPT Turing machines. The left probability in Eq.(22) is taken over the distribution of the random tapes input to ${G_R}^\dagger$, P, V, H, and E. The right probability in Eq.(22) is taken over the distribution of the random tapes input to G_R, H, E', and E. E' may use E as a black box.

Here, the definition includes the case where $\{(g'_i, m'_i)\}_{i=1,\cdots,k}$ is totally decrypted cipher-texts, that is, the case where $\{m'_i\}_{i=1,\cdots,k}$ are plain-texts.

If an argument for shuffle-decryption is CPH, we can say that for an input cipher-texts set $\{(g_i, m_i)\}$ and its corresponding output cipher-texts set $\{(g'_i, m'_i)\}$, whatever an honest verifier who has partial information $(H(I_n, enc(U), X_n, \phi))$ about the common input (X_n) and previously executed protocols (K) can learn $(f(\phi))$ about the permutation (ϕ) after interacting with a prover, can also - in an **overwhelming** number of cases of common input (X_n)- be efficiently computed from that common input (X_n) and that partial information $(H(I_n, enc(U), X_n, \phi))$ alone using a PPT Turing machine E' without interaction with the prover as long as the prover has chosen the private key x', permutation ϕ, and random numbers $\{s_i\}_{i=1,\cdots,k}$ uniformly and randomly.

Note that we are considering the case even where malicious and colluding players, who have the results of other shuffle-decryptions with the same x', are engaged in generating $\{(g_i, m_i)\}$ of common input. Hence, CPH guarantees security when shuffle-decryptions with the same private key are repeatedly executed.

---

†Since the probability is taken over a distribution containing x', we have excluded any adversary who knows x'.

Theorem 8: If the decision Diffie-Hellman problem is difficult to solve, the proposed argument for shuffle-decryption (P, V, G_R) is complete-permutation-hiding.

Proof. The proof is given in the appendix. □

6. Efficiency

In this section, we compare the efficiency of the proposed protocol described in Section 3 (Pro-I) and the proposed protocol described in Section 4 (Pro-II) to (FS), the protocol proposed in [8], (Fmmos), the protocol proposed in [7], and (Groth), the protocol proposed in [12]. We have assumed the lengths of p and q to be 1024 and 160.

Let us first compare them, in Table 1[†], by the number of exponentiations used in each protocol when the number of cipher-texts is k. "S-P" and "S-V" denote the number of exponentiations required for P and V to prove and verify a shuffle. 'SD-P" and "SD-V" denote the number of exponentiations required for P and V to prove and verify a shuffle-decryption. The numbers for (FS), (Fmmos), (Groth), and (Pro-I) are the sum of those required to prove a shuffle and those required to prove a decryption. The 3-round protocol for proving decryption is as follows.

i. Given shuffled cipher-texts $\{(g'_i, \bar{m}_i)\}_{i,\ldots,k}$ and shuffle-decrypted cipher-texts $\{(g'_i, m'_i)\}_{i,\ldots,k}$, a prover first chooses $\{\beta_i \in_R \mathbb{Z}/q\mathbb{Z}\}_{i=1,\ldots,k}$ uniformly and randomly and then generates and sends to a verifier:

$$\bar{g}_i = g_i'^{\beta_i} \bmod p.$$

ii. Next, the verifier chooses $\{c_i \in_R \mathbb{Z}/q\mathbb{Z}\}_{i\ldots,k}$ and sends it back to the prover.

iii. Next, the prover returns

$$r'_i = x' c_i + \beta_i \bmod q.$$

iv. Finally, the verifier checks that

$$g_i'^{r'_i} = (\bar{m}_i/m'_i)^{c_i} \bar{g}_i \bmod p$$

holds.

As is mentioned in Subsection 4.3, the technique of proving the correctness of both shuffling and decryption simultaneously is not directly available to the protocol proposed in [12] since η, which is a combination of $\{g'_j\}_{i=1,\cdots,k}$ and the challenge $\{c_i\}_{i=1,\cdots,k}$, does not appear in that protocol. And since a more efficient proof for decryption is not known, one of the most efficient ways for proving the correctness of both shuffling and decryption using [12] is to use the above presented proof for decryption.

If we adopt the computation tools described in [15], such as the simultaneous multiple exponentiation algorithm and the fixed-base comb method, the cost for computing exponentiations can be heuristically reduced. We estimated that multiple exponentiations cost a $1/3$ and exponentiations by the fixed-base comb method cost a $1/12$ (when the number of cipher-texts is large) of that of single exponentiation. Estimations done in this way are given in Table 2. Even if two protocols require the same number of exponentiations, one that can benefit from fixed-base comb method more than the other is able to reduce its computational cost more than the other.

Table 1 Numbers of exponentiations required in each protocol.

	FS	Fmmos	Groth	Pro-I	Pro-II
S-P	$8k$		$6k$	9k	
S-V	$10k$		$6k$	6k	
SD-P	$(9k)$	$9k$	$(7k)$	(10k)	$8k$
SD-V	$(12k)$	$10k$	$(8k)$	(8k))	$6k$

Table 2 Cost of computation required in each protocol.

	FS	Fmmos	Groth	Pro-I	Pro-II
S-P	$1.4k$		$1.75k$	1.75k	
S-V	$3.3k$		$1.75k$	2k	
SD-P	$(2.4k)$	$1.75k$	$(2.75k)$	(2.75k)	$1.9k$
SD-V	$(4.5k)$	$3.3k$	$(3k)$	(3.2k))	$2k$

Table 3 Communication bits required in each protocols.

	FS	Fmmos	Groth	Pro-I	Pro-II
S	$5044k$		$1184k$	1344k	
SD	$(6388k)$	$5044k$	$(2528k)$	(2688k)	$1344k$
rounds	3	5	7	3	5

Table 4 Security levels of each protocol.

	FS	Fmmos	Groth	Pro-I	Pro-II
soundness	DL	DL	DL	DL	DL
anonymity	CPH	CPH	PZK	PZK	CPH

Table 3 lists the number of communication bits and number of rounds required for protocols. "S" denotes the number of communication bits used when proving a shuffle, "SD" denotes the number of communication bits used when proving a shuffle-decryption, and "rounds" denotes the number of rounds required for protocols. The numbers for (FS), (Fmmos), (Groth), and (Pro-I) include intermediate state bits, i.e., those of shuffled cipher-texts.

Table 4 lists the level of security achieved by protocols. Here "DL" denotes that the corresponding protocol is sound as long as the discrete logarithm problem is difficult to solve. For Groth's protocol, we consider the case when it shuffles ElGamal cipher-texts. "CPH" denotes complete permutation hiding. "PZK" denotes perfect zero-knowledge. The protocol proposed in [7] can also be proved to be CPH. The protocol proposed in [8] can also be proved to be CPH assuming $y = 1$. The security of this protocol is proved also in [17]

Our protocols and the protocols of [7], [8] require a rather long public parameter F_k. Although the protocol of [12] also requires such a parameter, it can be reduced greatly at the cost of increasing the amount of both computation and communication.

From Tables 2, 3, and 4, we can conclude that our

[†]The computational cost required for provers of protocol "Pro-I" can be reduced by $1k$ if $q \bmod 3 = 2$.

zero-knowledge argument for shuffling is the most efficient among the less-than-7-round perfect zero-knowledge arguments for shuffling. The only comparable protocol is that of [22], which requires $640k$ exponentiations.

From Tables 2 and 3, we can conclude that computational complexity with our argument for shuffle-decryption protocol represents a 32% improvement in efficiency over that of (Groth)[12], while communication complexity improves by 47%. Our protocol requires two rounds less than that of Groth's [12].

7. Robustness

For the proposed argument for shuffle-decryption to be robust, we may consider that it is desirable to provide a method to make threshold shuffle-decryption. Since a shuffle and decryption are executed in succession by a single prover, it is not easy to make threshold shuffle-decryption. However, if each prover distributes his secret key to other provers in a desired threshold, robustness can be provided easily. In case any prover quits shuffle-decryption after some provers had shuffle-decrypted the cipher-texts, honest provers are able to collaborately recover the secret key of the malicious prover and make explicit decryption with that secret key. With this strategy, unless the number of honest provers is less than an appropriately configured threshold, honest provers are able to shuffle-decrypt all the cipher-texts. Since the cost for distributing secret keys depends on the number of provers but not on the number of cipher-texts, its cost is negligible when the number of cipher-texts is huge.

8. Conclusion

In this paper, I have proposed an efficient argument for shuffling and an efficient argument for shuffle-decryption. The former is most efficient in computational and in communication complexity among 3-move honest verifier perfect zero-knowledge arguments for shuffling of ElGamal cipher-texts. The latter protocol is the most efficient in computational, communication, and round complexity, as a whole, in proving the correctness of both shuffling and decryption of ElGamal cipher-texts. I also proposed a formal definition for the core requirement of unlinkability in arguments for shuffle-decryption, and then proved the latter protocol enjoys this property.

Acknowledgments

The author would like to thank Hiroaki Anada, Satoshi Obana, Seigo Arita, Kazue Sako, Tatsuaki Okamoto, and Eiichiro Fujisaki for their many helpful discussions.

References

[1] M. Abe, "Mix-networks on permutation networks," Advances in Cryptology—ASIACRYPT'99, LNCS 1716, pp.258–273, Springer-Verlag, 1999.
[2] S. Brands, "An efficient off-line electronic cash system based on the representation problem," CWI Technical Report CS-R9323, 1993.
[3] D. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," Commun. ACM, vol.24, no.2, pp.84–88, 1981.
[4] R. Cramer, I. Damgård, and B. Schoenmakers, "Proofs of partial knowledge and simplified design of witness hiding protocols," Crypto'94, LNCS 839, pp.174–187, 1994.
[5] R. Cramer and V. Shoup, "A practical key cryptosystem provably secure against adaptive chosen ciphertext attack," Advances in Cryptology—Crypto'98, LNCS 1462, pp.13–25, 1998.
[6] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," Advances in Cryptology—CRYPTO'86, LNCS 263, pp.186–194, 1986.
[7] J. Furukawa, K. Mori, S. Obana, and K. Sako, "An implementation of a universally verifiable electronic voting protocol based on shuffling," Financial Cryptography 2002.
[8] J. Furukawa and K. Sako, "An efficient protocol for proving a shuffle," Advances in Cryptology—CRYPTO 2001, LNCS 2139, pp.368–387, 2001.
[9] O. Goldreich, "A uniform-complexity treatment of encryption and zero-knowledge," J. Cryptol., vol.6, pp.21–53, 1993.
[10] S. Goldwasser and S. Micali, "Probabilistic encryption," J. Comput. Syst. Sci., vol.28, no.2, pp.270–299, 1984.
[11] P. Golle, S. Zhong, D. Boneh, M. Jakobsson, and A. Juels, "Optimistic mixing for exit-polls," Asiacrypt 2002, LNCS 2501, pp.451–465, 2002.
[12] J. Groth, "A verifiable secret shuffle of holomorphic encryptions," Public Key Cryptography—PKC 2003, LNCS 2567, pp.145–160, 2003.
[13] M. Jakobsson, "A practical mix," Eurocrypt'98, LNCS 1403, pp.448–461, 1998.
[14] A. Juels and M. Jakobsson, "An optimally robust hybrid mix network," Proc. 20th Annual ACM Symposium on Principles of Distributed Computation, 2001.
[15] A. Menezes, C. van Oorschot, and S. Vanstone, Handbook of Applied Cryptography, pp.617–627, CRC Press, 1997.
[16] C.A. Neff, "A verifiable secret shuffle and its application to E-voting," ACMCCS 01, pp.116–125, 2001.
[17] L. Nguyen, R. Safavi-Naini, and K. Kurosawa, "Verifiable shuffles: A formal model and a Paillier-based efficient construction with provable security," ACNS 2004, pp.61–75, 2004.
[18] M. Ohkubo and M .Abe, "A length-invariant hybrid mix," Asiacrypt 2000, LNCS 1976, pp.178–191, 2000.
[19] W. Ogata, K. Kurosawa, K. Sako, and K. Takatani, "Fault tolerant anonymous channel," ICICS, LNCS 1334, pp.440–444, 1997.
[20] C. Park, K. Itoh, and K. Kurosawa, "Efficient anonymous channel and all/nothing election protocol," Eurocrypt'93, pp.248–259, 1993.
[21] K. Sako, "Electronic voting protocols allowing open objection to the tally," IEICE Trans. Fundamentals, vol.E77-A, no.1, pp.24–30, Jan. 1994.
[22] K. Sako and J. Kilian, "Receipt-free mix-type voting protocol—A practical solution to the implementation of voting booth," Eurocrypt'95, LNCS 921, pp.393–403, 1995.
[23] C.P. Schnorr, "Efficient signature generation by smart cards," J. Cryptol., vol.4, pp.161–174, 1991.
[24] C.P. Schnorr and M. Jakobsson, "Security of signed ElGamal encryption," ASIACRYPT 2000, pp73–89, 2000.
[25] V. Shoup and R. Gennaro, "Securing threshold cryptosystems against chosen ciphertext attack," EUROCRYPT 1998, pp.1–16, 1998.

Appendix A: Security of the Proposed Zero-Knowledge Argument for Shuffle

A.1 Proof of Theorem 4

Lemma 1: If V accepts the protocol with a probability p, which implies, Eq. (9) holds, then

$$\prod_{\nu=-4}^{k} f_\nu{}^{r_\nu} = \prod_{\mu=0}^{k} f'_\mu{}^{c_\mu} \pmod p \qquad (\mathrm{A}\cdot 1)$$

$$\prod_{\nu=-4}^{k} f_\nu{}^{r'_\nu} = \tilde{f}'_0 \prod_{i=1}^{k} f'_i{}^{c_i{}^2} \pmod p \qquad (\mathrm{A}\cdot 2)$$

will hold with the probability at least $p - 1/2^{|q|}$.

Proof. From

$$\prod_{\nu=-4}^{k} f_\nu{}^{r_\nu+\alpha r'_\nu} = f'_0 \tilde{f}'_0{}^{\alpha} \prod_{\mu=-1}^{k} f'_i{}^{c_i+\alpha c_i{}^2} \pmod p$$

$$\Leftrightarrow \frac{\prod_{\nu=-4}^{k} f_\nu{}^{r_\nu}}{\prod_{\mu=0}^{k} f'_\mu{}^{c_\mu}} = \left(\frac{\tilde{f}'_0 \prod_{i=1}^{k} f'_i{}^{c_i{}^2}}{\prod_{\nu=-4}^{k} f_\nu{}^{r'_\nu}}\right)^{\alpha} \pmod p,$$

it is clear that the probability that V will choose α such that Eq. (9) holds is negligible if Eqs. (A·1) and (A·2) do not hold. □

Lemma 2: If V accepts the protocol with a probability p, then there is a polynomially bounded knowledge extractor K that can, within an expected number of steps bounded by $O(n/p)$, extract from P an $\{A_{\nu\mu}\}_{\nu=-4,\ldots,k;\mu=0,\ldots,k}$ which satisfies Eq. (7) and an $\{A'_\nu\}_{\nu=-4,\ldots,k}$ which satisfies Eq. (8).

Proof. Let us call every challenge $\{(c_\mu)_{\mu=0,\cdots,k}\}$ to which P can compute responses $\{(r_\nu)_{\nu=-4,\ldots,k}\}$ such that Eq. (9) holds for overwhelming probability as *acceptable challenge*. From Lemma 1, Eq. (A·1) also holds for these challenges. Suppose that K has chosen n linearly independent acceptable challenges. Then, K extracts, from P, n responses that corresponds to these challenges, from which K is able to extract an $\{A_{\nu\mu}\}_{\nu=-4,\ldots,k;\mu=0,\ldots,k}$ that satisfies the relation:

$$r_\nu \equiv \sum_{\mu=0}^{k} A_{\nu\mu} c_\mu \pmod q \quad \nu = -3, \cdots, k$$

for every $n+1$ challenge and response. Such an $\{A_{\nu\mu}\}_{\nu=-4,\ldots,k;\mu=0,\ldots,k}$ satisfies Eq. (7).

Now, we estimate the expected number of steps K is required to obtain these acceptable challenges. Suppose that K has obtained ℓ linearly independent acceptable challenges. If K tries to find other accepting challenges by randomly choosing challenges, the expected number of times K chooses challenges until it succeeds in finding an acceptable challenge is $1/p$. The probability that the obtained acceptable challenge is linearly independent to already obtained ℓ acceptable challenges is at least $1 - (|q|^\ell/p|q|^n)$, which is overwhelming. Therefore, with the expected number of steps K is required to obtains n linearly independent acceptable challenges is $O(n/p)$.

With respect to $\{A'_\nu\}_{\nu=-4,\ldots,k}$, the proof is similar. □

Lemma 3: Assume that there is a knowledge extractor K that can extract an $\{A_{\nu\mu}\}_{\nu=-4,\ldots,k;\mu=0,\ldots,k}$ and an $\{A'_\nu\}_{\nu=-4,\ldots,k}$ from P that satisfies Eq. (7) and (8). If there is also a knowledge extractor K' that can extract an $\{r_\nu, r'_\nu\}_{\nu=-4,\ldots,k}$ from P that satisfies Eq. (A·1) and (A·2) such that either

$$r_\nu \not\equiv \sum_{\mu=0}^{k} A_{\nu\mu} c_\mu \pmod q$$

$$\text{or} \quad r'_\nu \not\equiv \sum_{i=1}^{k} A_{\nu i} c_i{}^2 + A'_\nu \pmod q$$

for some $\nu = -4, \ldots, k$, then K' can generate non-trivial integers $\{a_\nu\}_{\nu=-4,\ldots,k}$ satisfying $\prod_{\nu=-4}^{k} f_\nu{}^{a_\nu} \equiv 1 \pmod p$ with overwhelming probability.

Proof. The following $\{f_\nu\}$-based expression will be a non-trivial representation of 1:

$$\prod_{\nu=-4}^{k} f_\nu{}^{\sum_{\mu=0}^{k} A_{\nu\mu} c_\mu - r_\nu} \equiv 1 \pmod p$$

$$\prod_{\nu=-4}^{k} f_\nu{}^{\sum_{i=1}^{k} A_{\nu i} c_i{}^2 + A'_\nu - r_\nu} \equiv 1 \pmod p$$

□

Lemma 4: Assume that $\{A_{\nu\mu}\}_{\nu=-4,\ldots,k;\mu=0,\ldots,k}$, $\{A_\nu\}_{\nu=-4,\ldots,k}$ and w are givens. Also assume that $\{r_i\}_{i=1,\ldots,k}$, r_{-2} and r'_{-3} will be generated, for a given $\{c_i\}_{i=1,\cdots,k}$, as

$$r_\nu = \sum_{\mu=0}^{k} A_{\nu\mu} c_\mu \bmod q \quad \nu = -3, 1, 2, \ldots, k$$

$$r'_{-3} = \sum_{i=1}^{k} A_{-3i} c_i{}^2 + A'_{-3} \bmod q$$

If the given $\{A_{ij}\}_{i=1,\ldots,k;j=1,\ldots,k}$ does not satisfy Eq. (1), then Eq. (12) will not hold with overwhelming probability for a uniformly and randomly chosen $\{c_i\}_{i=1,\cdots,k}$.

Proof. Eq. (12) is equivalent to

$$\sum_{i=1}^{k}\left(\sum_{h=1}^{k} A_{hi}A_{hi}A_{hi} - \delta_{iii}\right) c_i{}^3 + 3\sum_{i\neq j}\left(\sum_{h=1}^{k} A_{hi}A_{hi}A_{hj} - \delta_{iij}\right) c_i{}^2 c_j + 6\sum_{i>j>k}\left(\sum_{h=1}^{k} A_{hi}A_{hj}A_{hk} - \delta_{ijk}\right) c_i c_j c_k \equiv \sum_{i=1}^{k}\sum_{j=1}^{k}\left\{\left(\sum_{h=1}^{k} 3A_{h0}A_{hi}A_{hj} - A'_{-3i}\delta_{ij}\right)\right\} c_i c_j$$

$$+\sum_{i=1}^{k}\left\{\left(\sum_{h=1}^{k}3A_{h0}^{2}A_{hi}-A_{-2i}\right)\right\}c_i$$
$$+\left\{\left(\sum_{h=1}^{k}A_{h0}^{3}-A_{-20}-A'_{-30}\right)-w\right\}\pmod q.$$

The first three terms of the expression above and the fact that $2,3\not| q$ indicates that if Eq. (1) does not hold for some set $\{i,j,k\}$, then the probability will be negligible that Eq. (12) will hold for a uniformly and randomly chosen $\{c_i\}_{i=1,\cdots,k}$. □

Lemma 5: Assume that $\{A_{\nu\mu}\}_{\nu=-4,\ldots,k;\mu=0,\ldots,k}$ and $\dot{w}$ are givens. Also assume that $\{r_i\}_{i=1,\ldots,k}$ and r_{-4} will be generated for a given $\{c_i\}_{i=1,\cdots,k}$ as

$$r_\nu = \sum_{\mu=0}^{k}A_{\nu\mu}c_\mu \bmod q \quad \nu=-4,\ldots,k.$$

If the given $\{A_{\nu\mu}\}_{i=1,\ldots,k;j=1,\ldots,k}$ does not satisfy Eq. (2), then Eq. (13) will not hold with overwhelming probability for a uniformly and randomly chosen $\{c_i\}_{i=1,\cdots,k}$.

Proof. Because Eq. (13) is equivalent to

$$\sum_{i=1}^{k}\left(\sum_{h=1}^{k}A_{hi}A_{hi}-\delta_{ii}\right){c_i}^2$$
$$2\sum_{i>j}\left(\sum_{h=1}^{k}A_{hi}A_{hj}-\delta_{ij}\right)c_ic_j$$
$$+\sum_{i=1}^{k}\left(\sum_{h=1}^{k}2A_{h0}A_{hi}-A_{-4i}\right)c_i$$
$$+\sum_{h=1}^{k}A_{h0}^{2}-A_{-30}-\dot{w}\equiv 0 \pmod q.$$

□

Lemma 6: Assume that $\{g_\nu,m_\nu\}_{(\nu=0,\ldots,k)}$ and $\{A_{\nu\mu}\}_{(\nu=0,\ldots,k;\mu=0,\ldots,k)}$ are givens. Also assume that $\{r_\nu\}_{\nu=0,\ldots,k}$ will be generated for a given $\{c_i\}_{i=1,\cdots,k}$ as

$$r_\nu = \sum_{\mu=0}^{k}A_{\nu\mu}c_\mu \bmod q \quad \nu=0,\ldots,k.$$

If the equations

$$g'_\mu \equiv \prod_{\nu=0}^{k}{g_\nu}^{A_{\nu\mu}} \pmod p \quad \mu=0,\cdots,k \qquad (\text{A}\cdot 3)$$

$$m'_\mu \equiv \prod_{\nu=0}^{k}{m_\nu}^{A_{\nu\mu}} \pmod p \quad \mu=0,\cdots,k \qquad (\text{A}\cdot 4)$$

do not hold, then Eqs. (10) and (11) will not hold with overwhelming probability for a uniformly and randomly chosen $\{c_i\}_{i=1,\cdots,k}$.

Proof. Eq. (10) is equivalent to

$$\prod_{\mu=0}^{k}\left(\frac{\prod_{\nu=0}^{k}{g_\nu}^{A_{\nu\mu}}}{g'_\mu}\right)^{c_\mu}\equiv 1 \pmod p.$$

If Eq. (A· 3) does not hold, then Eq. (10) will not hold with overwhelming probability for a uniformly and randomly chosen $\{c_i\}_{i=1,\cdots,k}$.

By the same logic, if Eq. (A· 4) does not hold, then (11) will not hold with overwhelming probability for a uniformly and randomly chosen $\{c_i\}_{i=1,\cdots,k}$. □

*Proof.***(Theorem 4)**

From Lemmata 1, 2, and 3, there is a knowledge extractor K that can extract an $\{A_{\nu\mu}\}_{\nu=-4,\ldots,k;\mu=0,\ldots,k}$ and a $\{A'_\nu\}_{\nu=-4,\ldots,k}$ from P that respectively satisfy Eqs. (7) and (8) within an expected number of steps bounded by $O(n/p)$. Further, K can either, from values extracted from P, (1) only generate a response that satisfies equations

$$r_\nu \equiv \sum_{\mu=0}^{k}A_{\nu\mu}c_\mu \pmod q \quad \nu=-4,\ldots,k$$
$$r'_\nu \equiv \sum_{i=1}^{k}A_{\nu i}c_i + A'_\nu \pmod q \quad \nu=-4,\ldots,k$$

for a given challenge, or (2) generate non-trivial integers $\{a_\nu\}_{\nu=-4,\ldots,k}$ satisfying $\prod_{\nu=-4}^{k}f_\nu^{a_\nu}\equiv 1 \pmod p$ with overwhelming probability, with overwhelming probability. Note that (2) will happen with negligible probability as long as solving the discrete logarithm problem is difficult.

From Lemmata 4 and 5, the $\{A_{\nu\mu}\}_{(\nu=1,\ldots,k;\mu=1,\ldots,k)}$ that can be extracted by K from P satisfies Eqs. (1) and (2). From Theorem 1, sub-matrix $(A_{ij})_{i,j=1,\cdots,k}$ is a permutation matrix. From Lemma 6, the equations

$$g'_\mu \equiv \prod_{\nu=0}^{k}{g_\nu}^{A_{\nu\mu}} \pmod p \quad \mu=0,\cdots,k$$
$$m'_\mu \equiv \prod_{\nu=0}^{k}{m_\nu}^{A_{\nu\mu}} \pmod p \quad \mu=0,\cdots,k$$

hold for the $\{A_{\nu\mu}\}_{(\nu=0,\ldots,k;\mu=0,\ldots,k)}$ above.

Therefore, as long as solving the discrete logarithm problem is difficult, there is a knowledge extractor K that can extract an $\{A_{\nu i}\}_{(\nu=0,\ldots,k;i=1,\ldots,k)}$ from P satisfying Eq. (6) within an expected number of steps bounded by $O(n/p)$, and its sub-matrix $(A_{ij})_{i,j=1,\cdots,k}$ will be a permutation matrix if V accepts the protocol. □

A.2 Proof of the Theorem 5

Proof. Given p,q,g_0,m_0, $\{g_i,m_i\}_{i=1,\cdots,k}$, $\{g'_i,m'_i\}_{i=1,\cdots,k}$, $\{f_\nu\}_{(\nu=-4,\ldots,k)}$, a simulator uniformly and randomly chooses r_ν, r'_ν, c_i, f'_i for $\mu=0,\ldots,k;\nu=-4,\ldots,k$ and $i=1,\ldots,k$. It then generates $\tilde{f}'_0, f'_0, g'_0, m'_0, w$ and $\dot{w}$ so as to satisfy Eqs. (A· 1), (A· 2), (10), (11), (12), and (13). □

Appendix B: Security of the Proposed Verifiable Shuffle-Decryption

B.1 Proof of Theorem 7

Lemma 7: If V accepts the protocol with a probability p, i.e., Eq. (16) holds, then

$$\prod_{\nu=-2}^{k} f_\nu{}^{r_\nu} = \prod_{\mu=0}^{k} f'_\mu{}^{c_\mu} \pmod p \tag{A·5}$$

$$\prod_{\nu=-2}^{k} f_\nu{}^{r'_\nu} = \tilde{f}'_0 \prod_{i=1}^{k} f'_i{}^{c_i{}^2} \pmod p \tag{A·6}$$

will hold with the probability p.

Proof. From

$$\prod_{\nu=-2}^{k} f_\nu{}^{r_\nu+\alpha r'_\nu} = f'_0 \tilde{f}'_0{}^{\alpha} \prod_{\mu=-1}^{k} f'_i{}^{c_i+\alpha c_i{}^2} \pmod p$$

$$\Leftrightarrow \frac{\prod_{\nu=-2}^{k} f_\nu{}^{r_\nu}}{\prod_{\mu=0}^{k} f'_\mu{}^{c_\mu}} = \left(\frac{\tilde{f}'_0 \prod_{i=1}^{k} f'_i{}^{c_i{}^2}}{\prod_{\nu=-2}^{k} f_\nu{}^{r'_\nu}}\right)^{\alpha} \pmod p,$$

it is clear that the probability that V will choose α such that Eq. (16) holds is negligible if Eqs. (A·5) and (A·6) do not hold. □

Lemma 8: If V accepts the protocol with a probability p, then there is a knowledge extractor K that can, within an expected number of steps bounded by $O(n/p)$, extract an $\{A_{\nu\mu}\}_{\nu=-2,\ldots,k;\mu=0,\ldots,k}$ and $\{A'_\nu\}_{\nu=-2,\ldots,k}$ from P satisfying

$$f'_\mu = \prod_{\nu=-2}^{k} f_\nu{}^{A_{\nu\mu}} \pmod p \quad \mu = 0, \cdots, k \tag{A·7}$$

$$\tilde{f}'_0 = \prod_{\nu=-2}^{k} f_\nu{}^{A'_\nu} \pmod p \tag{A·8}$$

Proof. Eqs. (A·5) and (A·6) hold with a probability p. Hence, the lemma can be proved in the same way as Lemma 2. □

Lemma 9: Assume there is a knowledge extractor k that can extract $\{A_{\nu\mu}\}_{\nu=-2,\ldots,k;\mu=0,\ldots,k}$ and $\{A'_\nu\}_{\nu=-2,\ldots,k}$ from P satisfying Eqs. (A·7) and (A·8). If there is also a knowledge extractor K' that can extract an $\{r_\nu, r'_\nu\}_{\nu=-2,\cdots,k}$ that satisfies Eqs. (A·5) and (A·6) such that either

$$r_\nu \not\equiv \sum_{\mu=0}^{k} A_{\nu\mu} c_\mu \pmod q$$

$$\text{or } r'_\nu \not\equiv \sum_{i=1}^{k} A_{\nu i} c_i{}^2 + A'_\nu \pmod q$$

for some $\nu = -2, \ldots, k$, then K' can generate non-trivial integers $\{a_\nu\}_{\nu=-2,\cdots,k}$ satisfying $\prod_{\nu=-2}^{k} f_\nu{}^{a_\nu} = 1 \pmod p$ with overwhelming probability.

Proof. The following $\{f_\nu\}$-based expression will be a non-trivial representation of 1:

$$\prod_{\nu=-2}^{k} f_\nu{}^{\sum_{\mu=0}^{k} A_{\nu\mu} c_\mu - r_\nu} = 1 \pmod p$$

$$\prod_{\nu=-2}^{k} f_\nu{}^{\sum_{i=1}^{k} A_{\nu i} c_i{}^2 + A'_\nu - r'_\nu} = 1 \pmod p$$

□

Lemma 10: Assume $\{A_{\nu\mu}\}_{\nu=-2,\ldots,k;\mu=0,\ldots,k}$, $\{A'_\nu\}_{\nu=-2,\cdots,k}$ and w are givens. Also assume $\{r_\nu, r'_\nu\}_{\nu=-2,\cdots,k}$ will be generated for a given $\{c_i\}_{i=1,\cdots,k}$ as

$$r_\nu = \sum_{\mu=0}^{k} A_{\nu\mu} c_\mu \bmod q \quad \nu = -2, \cdots, k$$

$$r'_\nu = \sum_{i=1}^{k} A_{\nu i} c_i{}^2 + A'_\nu \bmod q \quad \nu = -2, \cdots, k.$$

If the given $\{A_{\nu\mu}\}_{\nu=-3,\ldots,k;\mu=0,\cdots,k}$ does not satisfy Eq. (1), then Eq. (19) will not hold with overwhelming probability for a uniformly and randomly chosen $\{c_i\}_{i=1,\cdots,k}$.

Proof. It can be shown in a way similar to that used for Lemma 4. □

Lemma 11: Assume $\{g_\nu\}_{\nu=0,\ldots,k}$ and $\{A_{\nu\mu}\}_{\nu=0,\ldots,k;\mu=0,\ldots,k}$ are givens. Also assume $\{r_\nu\}_{\nu=0,\ldots,k}$ will be generated for a given $\{c_i\}_{i=1,\cdots,k}$ as

$$r_\nu = \sum_{\mu=0}^{k} A_{\nu\mu} c_\mu \bmod q \quad \nu = 0, \ldots, k$$

If equation

$$g'_\mu = \prod_{\nu=0}^{k} g_\nu{}^{A_{\nu\mu}} \pmod p \quad \mu = 0, \cdots, k \tag{A·9}$$

does not hold, then Eq. (17) will not hold with overwhelming probability for a uniformly and randomly chosen $\{c_i\}_{i=1,\cdots,k}$.

Proof. Eq. (17) is equivalent to

$$\prod_{\mu=0}^{k} \left(\frac{\prod_{\nu=0}^{k} g_\nu{}^{A_{\nu\mu}}}{g'_\mu}\right)^{c_\mu} = 1 \pmod p.$$

If Eq. (A·9) does not hold, then Eq. (17) will not hold with overwhelming probability for a uniformly and randomly chosen $\{c_i\}_{i=1,\cdots,k}$. □

Lemma 12: The following three items hold:

i. If Eq. (20) holds with non-negligible probability, then there is a knowledge extractor K that can extract x' and β satisfying equation

$$y = g_0{}^{x'} \bmod p, \tag{A·10}$$

and Eq. (15).

ii. Assume that β and x' satisfy Eqs. (A· 10) and (15). If Eq. (20) holds, then

$$r' = c'x' + \beta \bmod q \tag{A· 11}$$

will hold.

iii. Assume β, x' and ζ are givens. Also assume r' will be generated for a given c' as

$$r' = c'x' + \beta \bmod q$$

If equations

$$\eta = \zeta^{x'} \quad \left(= \left(\prod_{i=1}^{k} g_i'^{c_i} \right)^{x'} \right) \pmod p$$
$$\eta' = \zeta^{\beta} \pmod p$$

do not hold, then Eq. (21) will not hold with overwhelming probability for a uniformly and randomly chosen c'.

Proof. The proof is straight-forward and, hence, has been omitted here. □

Lemma 13: Assume $\{g_\nu, m_\nu\}_{\nu=0,\cdots,k}$, $(A_{\nu\mu})_{\nu=0,\cdots,k;\mu=0,\cdots,k}$, $\{g'_j\}_{j=1,\cdots,k}$ and x' are givens. Also assume that $\{r_\nu\}_{\nu=0,\cdots,k}$ and η will be generated, for a given $\{c_i\}_{i=1,\cdots,k}$, as

$$r_\nu = \sum_{\mu=0}^{k} A_{\nu\mu} c_\mu \bmod q \quad \nu = 0, \ldots, k$$
$$\eta = \left(\prod_{j=1}^{k} g_j'^{c_j} \right)^{x'} \bmod p. \tag{A· 12}$$

If equation

$$m'_i = g_i'^{-x'} \prod_{\nu=0}^{k} m_\nu^{A_{\nu i}} \pmod p \quad \mu = 0, \cdots, k \tag{A· 13}$$

does not hold, then Eq. (18) will not hold with overwhelming probability for a uniformly and randomly chosen $\{c_i\}_{i=1,\cdots,k}$.

Proof. After substituting Eq. (A· 12), Eq. (18) will be equivalent to

$$\prod_{\nu=0}^{k} m_\nu^{r_\nu} = \left(\prod_{i=1}^{k} g_i'^{c_i} \right)^{x'} \prod_{\mu=0}^{k} m_\mu'^{c_\mu} \bmod p$$
$$\prod_{\nu=0}^{k} m_\nu^{r_\nu} = m'_0 \prod_{i=1}^{k} (g_i'^{x'} m'_i)^{c_i} \bmod p.$$

Thus, by the same logic as Lemma 11, if equation

$$g_i'^{x'} m'_i = \prod_{\nu=0}^{k} m_\nu^{A_{\nu i}} \bmod p \ ,$$

which is equivalent to Eq. (A· 13), does not hold, then Eq. (18) will not hold with overwhelming probability for a uniformly and randomly chosen $\{c_i\}_{i=1,\cdots,k}$. □

Proof.(**Theorem 7**)

From Lemmata 7, 8, and 9, there is a knowledge extractor K that can extract an $\{A_{\nu\mu}\}_{\nu=-2,\ldots,k;\mu=0,\ldots,k}$ and an $\{A'_\nu\}_{\nu=-2,\ldots,k}$ from P that will respectively satisfy Eqs. (A· 5) and (A· 6) within an expected number of steps bounded by $O(n/p)$. Further, K can either, from values extracted from P, (1) generate only a response which satisfies equations

$$r_\nu = \sum_{\mu=0}^{k} A_{\nu\mu} c_\mu \bmod q \quad \nu = -2, \ldots, k$$
$$r'_\nu = \sum_{i=1}^{k} A_{\nu i} {c_i}^2 + A'_\nu \bmod q \quad \nu = -2, \ldots, k$$

for a given challenge or (2) generate non-trivial integers $\{a_\nu\}_{\nu=-2,\ldots,k}$ satisfying $\prod_{\nu=-2}^{k} f_\nu^{a_\nu} = 1 \pmod p$ with overwhelming probability. Note that (2) will happen with negligible probability as long as solving the discrete logarithm problem is difficult.

From Lemma 10, the sub-matrix $(A_{ij})_{i,j=1,\cdots,k}$ that can be extracted by K from P satisfies Eq. (1). From the corollary of Theorem 1 and the fact $q \bmod 3 = 2$, sub-matrix $(A_{ij})_{i,j=1,\cdots,k}$ is a permutation matrix. From Lemma 11, the equation

$$g'_i = \prod_{\nu=0}^{k} g_\nu^{A_{\nu i}} \bmod p \quad i = 1, \cdots, k$$

holds for the above $\{A_{\nu i}\}_{\nu=0,\ldots,k;i=1,\ldots,k}$.

From Lemmas 12 and 13, the equation

$$m'_i = g_i'^{-x} \prod_{\nu=0}^{k} m_\nu^{A_{\nu i}} \bmod p \quad i = 1, \cdots, k$$

also holds for the above $\{A_{\nu i}\}_{\nu=0,\ldots,k;i=1,\ldots,k}$.

Therefore, as long as solving the discrete logarithm problem is difficult, if an honest verifier V accepts the protocol with non-negligible probability, there exists a polynomially bounded knowledge extractor K that can extract an $\{A_{\nu i}\}_{\nu=0,\ldots,k;i=1,\ldots,k}$ and x' from P that satisfies Eqs. (14) and $y = {g_0}^{x'} \pmod p$ within an expected number of steps bounded by $O(n/p)$, and its submatrix $(A_{ij})_{i,j=1,\cdots,k}$ will be a permutation matrix.

Therefore, the protocol is sound as long as solving the discrete logarithm problem is difficult. □

B.2 Proof of Theorem 8

Let us define an extended version of the decision Diffie-Hellman problem, DDH_k^m. For primes p, q such that $q|p-1$, let $\Theta_{mk}^{(pq)}$ be a set consisting of $(k+1) \times m$ elements of $\mathbb{G}_q$ as

$$\Theta_{mk}^{(pq)} = (\Theta_{i\nu})_{i=1,\ldots,m;\nu=0,\ldots,k} \ ,$$

and let R_k^m be a set consisting of all different $\Theta_{mk}^{(pq)}$, where the

number of whose elements is $q^{m(k+1)}$.

Let D_k^m be the subset of R_k^m consisting of all the elements $\Theta_{mk}^{(pq)}$ of R_k^m that satisfy

$$\log_{\Theta_{10}} \Theta_{j0} = \log_{\Theta_{1\nu}} \Theta_{j\nu} \pmod q$$

for $\nu = 1, \ldots, k; j = 2, \ldots, m$, and the number of whose elements is $q^{(k+m)}$.

Definition 5: Given a $\Theta_{mk}^{(pq)}$ chosen uniformly and randomly from either a set R_k^m or a set D_k^m, the DDH_k^m problem is that of deciding whether the given $\Theta_{mk}^{(pq)}$ has been chosen from D_k^m or R_k^m.

The DDH_2^2 problem is a decision Diffie-Hellman problem.

Lemma 14: Solving the DDH_k^m problem is as difficult as solving a decision Diffie-Hellman problem. That is,

$$\begin{aligned}&\forall D\, \forall c > 0\, \exists N\, \forall n > N \\ &\quad |\Pr(D[\Theta_{mk}^{(pq)} \in_R D_k^m] = 1) \\ &\quad - \Pr(D[\Theta_{mk}^{(pq)} \in_R R_k^m] = 1)| \\ &\leq \frac{1}{n^c}\end{aligned}$$

holds if and only if the decision Diffie-Hellman problem is difficult to solve, where D is a polynomially bounded algorithm.

Proof. The lemma can be proved by a hybrid argument. (ref. Lemmas 1 and 2 in [8].) □

Let us next define a simulator S of the proposed protocol (P, V)[†]. We will show that if the protocol is not CPH, there will be a distinguisher that can distinguish the distribution of any simulated view from that of a real protocol view, where probability is taken over "the random tape of G_R" as well as other random tapes. And we then show that distinguishing them is equivalent to solving the decision Diffie-Hellman problem.

Definition 6: Given X_n, simulator S uniformly and randomly generates
$\{r_\nu, r'_\nu\}_{\nu=-2,\cdots,k}, \{c_i\}_{i=1,\cdots,k}, r', c' \in_R \mathbb{Z}/q\mathbb{Z}$, $\{f'_i \in_R \mathbb{G}_q\}_{i=1,\cdots,k}$, and $\eta \in_R \mathbb{G}_q$. It then generates:

$$f'_0 = \prod_{\nu=-2}^{k} f_\nu^{r_\nu} \prod_{i=1}^{k} f_i'^{-c_i} \bmod p$$

$$\tilde{f}'_0 = \prod_{\nu=-2}^{k} f_\nu^{r'_\nu} \prod_{i=1}^{k} f_i'^{-c_i^2} \bmod p$$

$$g'_0 = \prod_{\nu=0}^{k} g_\nu^{r_\nu} \prod_{i=1}^{k} g_i'^{-c_i} \bmod p$$

$$m'_0 = \eta^{-1} \prod_{\nu=0}^{k} m_\nu^{r_\nu} \prod_{i=1}^{k} m_i'^{-c_i} \bmod p$$

$$w = \sum_{j=1}^{k} (r_j^3 - c_j^3) - r_{-2} - r'_{-1} \bmod q$$

$$y' = g_0^{r'} y^{-c'} \bmod p$$

$$\eta' = \zeta^{r'} \eta^{-c'} \bmod p.$$

S outputs:

$$\begin{aligned}&p, q, y, g_0, m_0, \{(g_i, m_i)\}_{i=1,\ldots,k}, \{(g'_i, m'_i)\}_{i=1,\ldots,k}, \\ &\{f_\nu\}_{\nu=-2,\ldots,k}, f'_0, \{f'_i\}_{i=1,\ldots,k}, \tilde{f}'_0, \\ &g'_0, m'_0, w, \{c_i\}_{i=1,\ldots,k}, \\ &\{r_\nu\}_{\nu=-2,\ldots,k}, \{r'_\nu\}_{\nu=-2,\ldots,k}, \eta, \eta', y', c', r'.\end{aligned}$$

Let us next define an algorithm M which generates distribution of view from $\Theta_{2k}^{(pq)}$.

Definition 7: (**Algorithm** M.) Given $I_n = \{1^n, p, q, \bar{x}, \{M_i\}_{i=1,\cdots,k}, Z_n\}$, $enc(U)$, and $\Theta_{3k}^{(pq)}$, M uniformly and randomly chooses $f_{-2}, f_{-1}, \{f_\nu\}_{i=1,\cdots,k}$ and generates

$$f_0 = \Theta_{30}$$
$$(g_i, m_i) = U(I_n, g_0, y, F_k) \in (\mathbb{G}_q \times \mathbb{G}_q)$$

where $F_k = \{f_\nu\}_{\nu=-2,\ldots,k}$. M then extracts $\{\bar{r}_i\}_{i=1,\cdots,k}$ such that $g_i = g_0^{\bar{r}_i} \bmod p$ using U as a black box. M uniformly and randomly chooses
$\{c_i\}_{i=1,\cdots,k}, \{r_\nu, r'_\nu\}_{\nu=-3,\cdots,k}, r', c' \in_R \mathbb{Z}/q\mathbb{Z}$, and a $k \times k$ permutation matrix $(A_{ji})_{i,j=1,\cdots,k}$. Then, for $i = 1, \ldots, k$, it generates:

$$g_0 = \Theta_{10}\ ,\quad y = \Theta_{20}\ ,\quad m_0 = y g_0^{\bar{x}} \bmod p\ ,$$

$$g'_i = \Theta_{1i} \prod_{j=1}^{k} g_j^{A_{ji}} \bmod p$$

$$m'_i = (y^{-\sum_{j=1}^{k} \bar{r}_j A_{ji}}) \Theta_{1i}^{\bar{x}} \prod_{j=1}^{k} m_j^{A_{ji}} \bmod p\ ,$$

$$A_{i0} = r_i - \sum_{j=1}^{k} A_{ij} c_j \bmod q$$

$$A_{-1i} = \sum_{j=1}^{k} 3 A_{j0} A_{ji} \bmod q$$

$$A_{-2i} = \sum_{j=1}^{k} 3 A_{j0}^2 A_{ij} \bmod q\ ,$$

$$\zeta = \prod_{i=1}^{k} g_i'^{c_i} \bmod p\ ,$$

$$\eta = y^{\sum_{i,j=1}^{k} \bar{r}_j A_{ji} c_i} \prod_{i=1}^{k} \Theta_{2i}^{c_i} \bmod p$$

$$\eta' = \zeta^{r'} \eta^{-c'} \bmod p$$

[†]If we want to prove that the protocol is computational zero-knowledge, we must show the existence of a simulator whose output is, even to a distinguisher who knows x', indistinguishable from the actual interaction. This is impossible here; i.e., the above protocol is not zero-knowledge. Though a distinguisher who knows x' is able to distinguish a real protocol-transcript from the output of this simulator (defined below), we are able to prove, by using this simulator, that the protocol hides permutations completely.

$$g'_0 = \zeta^{-1} \prod_{\nu=0}^{k} g_\nu{}^{r_\nu} \bmod p$$
$$m'_0 = \eta^{-1} \prod_{\nu=0}^{k} m_\nu{}^{r_\nu} \prod_{j=1}^{k} m'_j{}^{-c_j} \bmod p$$
$$y' = g_0{}^{r'} y^{-c'} \bmod p \ ,$$
$$f'_i = f_{-2}^{A_{-2i}} f_{-1}^{A_{-1i}} \Theta_{3i} \prod_{j=1}^{k} f_j{}^{A_{ji}} \bmod p$$
$$f'_0 = \prod_{\nu=-2}^{k} f_\nu{}^{r_\nu} \prod_{j=1}^{k} f'_j{}^{-c_j} \bmod p$$
$$\tilde{f}'_0 = \prod_{\nu=-2}^{k} f_\nu{}^{r'_\nu} \prod_{j=1}^{k} f'_j{}^{-c_j{}^2} \bmod p$$
$$w = \sum_{j=1}^{k} (r_j^3 - c_j^3) - r_{-2} - r'_{-1} \pmod q$$

M outputs:

$$p, q, y, g_0, m_0, \{(g_i, m_i)\}_{i=1,\ldots,k}, \{(g'_i, m'_i)\}_{i=1,\ldots,k},$$
$$\{f_\nu\}_{\nu=-2,\ldots,k}, f'_0, \{f'_i\}_{i=1,\ldots,k},$$
$$\tilde{f}'_0, g'_0, m'_0, w, \{c_i\}_{i=1,\ldots,k},$$
$$\{r_\nu\}_{\nu=-2,\ldots,k}, \{r'_\nu\}_{\nu=-2,\ldots,k}, \eta, \eta', y', c', r'.$$

as $View^\dagger(I_n, U, \Theta_{3k}^{(pq)})$. $View^\dagger$ is an abbreviation of $View^\dagger(I_n, U, \Theta_{3k}^{(pq)})$. M also outputs (A_{ji}) as $(A_{ji})^\dagger_{i,j=1,\cdots,k}$.

Lemma 15: Assume X_n is the output of G_R, whose input is a set
$I_n = \{1^n, p, q, \bar{x}, \{M_i\}_{i=1,\cdots,k}, Z_n\}$ and an $enc(U)$ Also assume $View_V^P$ is a view of (P, V) with respect to F_k, the above $enc(U)$, I_n and uniformly and randomly chosen g_0, y. If $\Theta_{3k}^{(pq)} \in_R D_k^3$, then the distribution of $View^\dagger$ for the above I_n and $enc(U)$ is equivalent to that of the above $View_V^P$ for the above I_n and $enc(U)$, where the distribution of $View^\dagger$ is considered with respect to
$f_{-2}, f_{-1}, \{f_i, c_i\}_{i=1,\cdots,k}, \{r_\nu, r'_\nu\}_{\nu=-2,\ldots,k}, r', c'$,
$(A_{ji})_{i,j=1,\cdots,k}$ and $\Theta_{3k}^{(pq)}$ and the distribution of $View_V^P$ is considered with respect to the random tapes input to G_R, P, and V and with respect to F_k.

Proof. The proof is clear from the following correspondences (for $i = 1, \ldots, k$):

$$\Theta_{10} \Leftrightarrow g_0, \Theta_{1i} \Leftrightarrow g_0^{A_{0i}},$$
$$\Theta_{20} \Leftrightarrow y, \Theta_{2i} \Leftrightarrow y^{A_{0i}},$$
$$\Theta_{30} \Leftrightarrow f_0, \Theta_{3i} \Leftrightarrow f_0^{A_{0i}}$$

□

Lemma 16: Assume X_n is the output of G_R, whose input is a set
$I_n = \{1^n, p, q, \bar{x}, \{M_i\}_{i=1,\cdots,k}\}$ and $enc(U)$. Also assume $View^*$ is an output of Simulator S whose input is X_n. If $\Theta_{3k}^{(pq)} \in_R R_k^3$, then the distribution of $View^\dagger$ for the above I_n and $enc(U)$ is equivalent to that of the above $View^*$ for the above I_n, where the distribution of $View^\dagger$ is considered with respect to
$f_{-2}, f_{-1}, \{f_i, c_i\}_{i=1,\cdots,k}, \{r_\nu, r'_\nu\}_{\nu=-2,\ldots,k}, r', c'$,
$(A_{ji})_{i,j=1,\cdots,k}$ and $\Theta_{3k}^{(pq)}$ and the distribution of $View^*$ is considered with respect to the random tapes input to G_R, S, and M and with respect to F_k.

Proof. In $View^\dagger$, there is sufficient randomness, originating from the elements in $\Theta_{3k}^{(pq)}$, to match the randomness of a shuffle and that of $f'_i, \eta \in_R \mathbb{G}_q$ uniformly and randomly chosen by the simulator. □

Lemma 17: There exists a PPT Turing machine K such that its output distribution on input of $I_n, g_0, y, enc(U), F_k, \phi$ i.e.,$K(I_n, g_0, y, enc(U), \phi)$ is the same as that of $View_V^P$.

Proof. Since $\bar{r}_i$ such that $g_i = g_0{}^{\bar{r}_i} \bmod p$ is extractable by the PPT Turing machine, K is able to generate m'_i and η as

$$m'_i = g'_i{}^{-x'} \prod_{\nu=0}^{k} m_\nu{}^{A_{\nu i}}$$
$$= y^{A_{0i}} y^{\sum_{j=1}^{k} \bar{r}_j A_{ji}} \prod_{\nu=0}^{k} m_\nu{}^{A_{\nu i}} \bmod p$$
$$\eta = \prod_{i=1}^{k} \left(y^{-A_{0i}} y^{-\sum_{j=1}^{k} \bar{r}_j A_{ji}} \right)^{c_i} \bmod p$$

without the knowledge of x'. Other variables in $View_V^P$ are perfectly simulatable from the above values. □

Proof.(**Theorem 8**)

We are able to prove that the contraposition of the first statement of the theorem holds. That is, we are able to prove that, if the negation of Formula (22) holds, then there exists a polynomially bounded algorithm that can solve the DDH_k^3 problem.

Assume that the negation of Formula (22), that is,

$$\forall E'^E \exists E \exists H \exists f \exists U \exists c > 0 \, \forall N \exists n > N \exists I_n$$
$$\Pr[E(View_V^P, H(I_n, enc(U), X_n, (A_{ji}))) = f((A_{ji}))]$$
$$\geq \Pr[E'^E(X_n, H(I_n, enc(U), X_n, (A_{ji}))) = f((A_{ji}))]$$
$$+ \frac{1}{n^c},$$

holds. If we let E' be an algorithm that first generates a simulated view $View'^P_V$ from X_n using the simulator defined in Definition 6, and then guess the permutation using E as a black box, then the formula

$$\exists E \exists U \exists c > 0 \, \forall N \exists n > N \exists I_n$$
$$\left| \Pr[E(View_V^P, H(I_n, enc(U), X_n, (A_{ji}))) = f((A_{ji}))] \right.$$
$$\left. - \Pr[E(View'^P_V, H(I_n, enc(U), X_n, (A_{ji}))) = f((A_{ji}))] \right|$$
$$\geq \frac{1}{n^c},$$

holds. If we include the definitions of H, f and E into E', then the formula

$$\exists_{E'}\,\exists_U\,\exists_{c>0}\,\forall_N\,\exists_{n>N}\,\exists_{I_n}$$
$$\left|\Pr[E'(View_V^P, I_n, enc(U), (A_{ji})) = 1]\right.$$
$$\left. - \Pr[E'(View'^P_V, I_n, enc(U), (A_{ji})) = 1]\right| \geq \frac{1}{n^c}$$

holds. From Lemma 15 the distribution of $\{View_V^P, I_n, enc(U), (A_{ji})\}$ is equivalent to that of $\{View^\dagger, I_n, enc(U), (A_{ji})^\dagger\}$ when $\Theta_{3k}^{(pq)} \in_R D_k^3$. From Lemma 16 the distribution of $\{View'^P_V, I_n, enc(U), (A_{ji})\}$ is equivalent to that of $\{View^\dagger, I_n, enc(U), (A_{ji})^\dagger\}$ when $\Theta_{3k}^{(pq)} \in_R R_k^3$. Therefore, we can construct a distinguisher D by using algorithms M and E as black boxes such that the formula

$$\exists_D\,\exists_{c>0}\,\forall_N\,\exists_{n>N}$$
$$|\Pr(D[\Theta_{3k}^{(pq)} \in_R D_k^3] = 1)$$
$$- \Pr(D[\Theta_{3k}^{(pq)} \in_R R_k^3] = 1)| \geq \frac{1}{n^c}$$

holds. This means that the DDH_k^3 problem can be solved by using E. Therefore, the contraposition of the lemma has been proved.

The second statement is satisfied from Lemma 17 □

A complete description of distinguisher D is as follows:

Definition 8: Given $\Theta_{3k}^{(pq)}$ of size n, D first chooses F_k uniformly and randomly, and chooses arbitrary $\{M_i\}_{i=1,\cdots,k}$, and $\mathbb{Z}/q\mathbb{Z}$ elements $\bar{x}$, Z, and $enc(U)$. It gives these values $(I_n, enc(U))$ and $\Theta_{3k}^{(pq)}$ to M, and then obtains $View^\dagger$, and permutation matrix $(A_{ji})^\dagger$ from M.

Next, D gives $View^\dagger, enc(U), I_n$, and $(A_{ji})^\dagger$ to the adversary E' and obtains response 1 or 0. Finally, D outputs the response of E'.

□

Appendix C: Proof of Theorem 1 Given in [8]

Proof. ($\Rightarrow$) is trivial. ($\Leftarrow$); Let us first show that there is exactly one non-zero element in each row vector of (A_{ij}) and then show the same for each column vector. Let C_i be an i-th column vector of matrix $(A_{ij})_{i,j=1,\ldots,k}$. From Eq. (2), $\sum_{h=1}^k (A_{hi}A_{hi})(A_{hg}) = \delta_{ig}$. Thus, $rank(A_{ij}) = k$, i.e., there is at least one non-zero element in each row and each column. Next, let us consider a vector $C_i \odot C_j (i \neq j)$ for which the operator $\odot$ is defined as $(a_1 \ldots a_k)\odot(b_1 \ldots b_k) = (a_1 b_1 \ldots a_k b_k)$. Define a vector $\hat{C} = \sum_{l=0}^k \kappa_l C_l$ for an arbitrary κ_l. From the facts that $(\hat{C}, C_i \odot C_j) = \sum_{l=1} \kappa_l \delta_{lij} = 0$ and that linear combinations of $\{C_l\}$ generate the space $(\mathbb{Z}/q\mathbb{Z})^k$, we obtain $C_i \odot C_j = \vec{0}$. This means that for any h, i and j such that $i \neq j$, either $A_{hi} = 0$ or $A_{hj} = 0$. Therefore, the number of non-zero elements in each row vector of (A_{ij}) is at most 1, and thus exactly 1.

From the above observations, matrix (A_{ij}) contains exactly k non-zero elements. Since $C_i \neq \vec{0}$ for all i, the number of non-zero elements in each column vector is also 1. Thus, there is exactly one non-zero element in each row vector and each column vector of matrix (A_{ij}) if Eqs. (2) and (1) hold.

From Eq. (2), the unique non-zero element e_i in the $i-th$ row must satisfy $e_i^2 = 1 \bmod q$, and from Eq. (1), it must satisfy $e_i^3 = 1 \bmod q$. This leads to $e_i = 1$ and to the fact that matrix $(A_{ij})_{i,j=1,\ldots,k}$ is a permutation matrix over $\mathbb{Z}/q\mathbb{Z}$. □

Jun Furukawa received master's degree in physics from Tokyo University in 1996, left the doctor course and joined NEC in 1999. His research interests are in cryptography.