PAPER *Special Section on Cryptography and Information Security*

Flexible-Routing Anonymous Networks Using Optimal Length of Ciphertext

Koji CHIDA[†a)] ***and*** **Masayuki ABE**[†b)], ***Members***

SUMMARY We present an efficient Hybrid Mix scheme that provides both routing flexibility and the optimal length of ciphertext. Although it is rather easy to embed routing information in the ciphertext, and a scheme that provides the optimal length of ciphertext is already known, it is not a trivial task to achieve both properties all at the same time. A critical obstacle for providing the optimal length of ciphertext is the session-key encapsulation header in a ciphertext that carries the encrypted session-key to each router, which linearly increases according to the number of intermediate routers. We solve this problem by improving the previously reported Hybrid Mix scheme such that the resulting scheme benefits from routing flexibility with a constant length of such headers. Our basic scheme is only secure against honest, but curious intermediate routers. Therefore, we further address the robustness issue to prevent malicious behavior by incorporating and improving an existing efficient approach based on the Message Authentication Code.

key words: *Mix-net, Hybrid Mix, Onion Routing, anonymity*

1. Introduction

1.1 Background

Since the invention of the secure anonymous channel by Chaum in 1981 [2], many studies have been proposed in order to improve the availability and security of the original scheme. The original scheme, called Chaum's *Mix-net*, comprises a series of servers to which a list of ciphertexts is input and a list of corresponding messages is output in a random order. Originally, Mix-net is a network scheme that provides anonymity among a certain group under general cryptographic assumptions.

Until now, various anonymous applications using Mix-net have been proposed, for instance, anonymous E-mail [10], anonymous Web browsing [6], anonymous peer-to-peer communications [7], and of course anonymous voting typified by [13], [14]. Some studies of Mix-net employ both a public-key algorithm (key-exchange mechanism) and symmetric-key encryption such as the Secure Socket Layer (SSL) in order to handle efficiently messages of various lengths. This hybrid type of Mix-net, called *Hybrid Mix*, is much needed to achieve efficient anonymous communications in E-mail, Web browsing, etc.

The basic style of Hybrid Mix such as in [15] involves a series of public-key encryptions to hide symmetric-keys. In addition, the Hybrid Mix scheme presented by Syverson et al. [15], called *Onion Routing*, provides routing flexibility. Here, routing flexibility means a property in which a user (or a control system) can select, relay to servers, and request from the servers message delivery. This aspect of routing flexibility is superior in terms of the privacy of the route, fault tolerance, and scalability. We briefly describe the Onion Routing scheme. Any anonymous user generates appropriate ciphertexts in layers and then the i-th relay server receives one of the ciphertexts, $C_{i-1} = \{\mathcal{P}_{PK_i}(K_{i,j})\|\mathcal{E}_{K_{i,j}}(\mathsf{Addr}_{i+1}\|C_i)\}$, from the $(i-1)$-st relay server, where $\mathcal{P}$ and PK_i are respectively a public-key encryption algorithm and the public-key of the i-th relay server, $\mathcal{E}$ and $K_{i,j}$ are respectively a symmetric-key encryption algorithm and the symmetric-key between the i-th relay server and an anonymous user represented by j, and Addr_{i+1} is the $(i+1)$-st server's address. Then, the i-th relay server sends C_i to the $(i+1)$-st relay server according to Addr_{i+1}. Note that, the m-th relay server receives $C_{m-1} = \{\mathcal{P}_{PK_m}(K_{m,j})\|\mathcal{E}_{K_{m,j}}(msg)\}$ if it is the last relay server, where msg is a message from an anonymous user. This aspect of routing flexibility is superior in terms of the privacy of the route, fault tolerance, and scalability. In particular, a flexible-routing Mix-net can be used as a scheme that is operated in an asynchronous mode assuming that monitoring of a certain network is intractable. Table 1 compares characteristics of synchronous/asynchronous modes of Mix-net. Note that, anonymity of an asynchronous mode is weaker than that of a synchronous mode because any synchronous Mix-net strengthens anonymity by shuffling multiple messages. Details are described in Sec. 1.3.

It is clear, however, that the resulting Onion Routing scheme increases the length of the ciphertext proportionally to the value of the product of the public-key size and the number of relay servers selected by the anonymous user. This is because there exist m session-key encapsulation headers, $\mathcal{P}_{PK_i}(K_{i,j})$, in ciphertext C_0, where m is the number of relay servers. This increased length causes not only loss of scalability, but also increases the number of input ciphertexts to relay to servers in the case that a user's ciphertext is divided into a certain size to prevent an attack by traffic analysis*. Therefore, to improve scalability and efficiency, it is desirable that the length of the ciphertext be as short as

Manuscript received March 22, 2004.
Manuscript revised June 25, 2004.
Final manuscript received August 31, 2004.

†The authors are with NTT Information Sharing Platform Laboratories, NTT Corporation, Yokosuka-shi, 239-0847 Japan.
a) E-mail: chida@isl.ntt.co.jp
b) E-mail: abe@isl.ntt.co.jp

*In fact, the Type 2 system of *Mixmaster Remailer* [10] realizes that all user's ciphertexts are separately sent to appropriate relay servers after being segmented into a specified size and the last relay server finally reassembles the message using the segments.

Table 1 Characteristics of synchronous/asynchronous modes of Mix-net.

	anonymity	delay time	application
synchronous (Chaum's Mix-net type)	strong	high	voting
asynchronous (Onion Routing type)	moderate	low	E-mail, P2P, Web browsing

possible.

In a recent study, Ohkubo and Abe [11] enhanced the Hybrid Mix scheme employing not a public-key encryption for key-delivery, but the technique of Diffie-Hellman key-exchange [4]. Due to this alteration in the key-sharing scheme, the length of the session-key encapsulation header becomes independent of the number of servers. This means that their scheme provides the *optimal length of ciphertext*. In addition, they indicated how to obtain robustness in their scheme following the concept described in [3]. The resulting Hybrid Mix is assured to return a correct result and to provide anonymity in the presence of corrupt servers.

There still remain, however, some issues with the scheme in [11]. The greatest obstacle is the lack of routing flexibility. This means that servers in the middle between a sender and the destination point are deployed sequentially and statically; therefore, the privacy of the route, fault tolerance, and scalability are lost. Another flaw is that their scheme provides robustness to a weaker model. More precisely, their scheme only achieves robustness against at most $\sqrt{m}$ corrupt server managers out of the total of m managers. This is because each layer of decryption is performed by a quorum of distributed managers. Therefore, their scheme is not assured to provide anonymity in the presence of more than $\sqrt{m}$ corrupt server managers.

Jakobsson and Juels [9] presented a scheme improving on that described in [11]. It is robust against any coalition of fewer than $m/2$ server managers. This means that their scheme provides *optimal robustness*. Recently, Abe and Imai pointed out the weakness in terms of anonymity of the scheme presented in [9], and, in the same paper, presented a way to repair the weakness [1]. However, the issue of routing flexibility still remains.

1.2 Our Contribution

We propose an efficient flexible-routing Hybrid Mix scheme with the optimal length of ciphertext, i.e., the session-key encapsulation header in the proposed scheme has a constant length. This efficiency improvement is not only an asymptotic argument, but it has a certain practical impact. For instance, with a standard setting of 2048-bit modulus RSA key-delivery or similar Diffie-Hellman key-exchange with five routers, a session-key encapsulation header is about 10k bits long in the naive solution while our solution reduces it to 2k bits. Obviously, the length can further be improved by using elliptic-curve techniques. In addition, we extend the proposed basic scheme to a scheme that enables the return of a message from a receiver who received an anonymous message to the anonymous sender.

We discuss the security, in terms of anonymity against colluding and passive, i.e., honest-but-curious, servers and users by showing the reduction to the *Gap Diffie-Hellman Problem* [12] and the *Matching Find-Guess Problem* [11], which addresses a general problem about indistinguishability of any symmetric-key encryption scheme. In addition, we introduce an approach to strengthen the basic scheme against active attacks, i.e., malicious servers and users. The resulting scheme is robust when the majority of the relay servers are uncorrupted. Our approach is based on that of Jakobsson and Juels [9], which uses Message Authentication Codes (MACs) as a kind of evidence of correct behavior. Since their scheme has a security problem, we incorporate the idea with modification following the suggestion by Abe and Imai [1].

1.3 Communication Model

The synchronous mode of anonymous communications such as Chaum's Mix-net aims to establish anonymous communications against a strong adversary who can observe the entire network. This type of Mix-net provides anonymity among a certain group in the presence of the adversary under general cryptographic assumptions. However, it is not suitable for real-time communication applications because of the shuffling of multiple messages.

On the other hand, if the ability of an adversary to observe the network is partially restricted, the asynchronous mode such as Onion Routing can provide real-time anonymous communications due to the routing flexibility. Note that, the flexible-routing scheme provides privacy of the route so the scheme is not suitable for a network with only a few destination points (e.g., the network has only a few anonymous service sites).

The proposed scheme is applicable to both synchronous and asynchronous modes, but in this paper we focus on the "semi"-asynchronous mode for providing both strong anonymity and fast anonymous communications. Here, the semi-asynchronous mode means that every relay server outputs either two messages with shuffling by storing the previous input ciphertext or only the stored message at specific times. If there are a lot of traffic and destination points in the network, this semi-asynchronous mode can be practical for real-time anonymous communications. This operation makes $2^{m/2}$ possible routes for any user on average using m relay servers, which are set between a sender and the destination point, if the probability that two or more messages accumulate at a relay server by a certain time is $1/2$. Note that, the semi-asynchronous mode does not lose anonymity even if the destination points are same due to the shuffling of the two messages.

Throughout this paper, following [10], we assume that a user's message is divided into a certain size to prevent an attack by traffic analysis.

2. Our Scheme

In this section, we first present a new *compressed key scheduling* protocol. This compressed key scheduling is a generalization of the method introduced in [11] and is named in [9]. Roughly speaking, it is a way of exchanging the common key between the user and each relay server while maintaining anonymity and the optimal length of ciphertext. Next, we present the full construction of the proposed basic Hybrid Mix scheme. After that, we estimate the level of computation/communication efficiency of the proposed scheme in addition to that of the trivial Hybrid Mix scheme. Finally, we discuss the security of the proposed scheme against some attacks.

Throughout this paper, we denote servers and users as M_i $(i = 1, \ldots, m)$ and U_j $(j = 1, \ldots, n)$, respectively. We assume that each M_i's address Addr_i is published on a bulletin board. Let $\mathcal{G}$ be a discrete logarithm instance generator such that $(p, q, g) \leftarrow \mathcal{G}(1^\rho)$, where p and q are primes such that $q \mid p - 1$ holds, g is an element of $(\mathbb{Z}/p\mathbb{Z})^*$ whose order is q, and ρ is a security parameter. We denote a unique subgroup of $(\mathbb{Z}/p\mathbb{Z})^*$ generated by g as $\langle g \rangle$. The following symbols $\mathcal{E}$, $\mathcal{D}$, $\mathcal{K}$, $\mathcal{M}$, and C are assumed to be a symmetric encryption scheme, where $\mathcal{E}$ and $\mathcal{D}$ are the encryption and decryption algorithms and $\mathcal{K}$, $\mathcal{M}$, and C are the spaces for keys, messages, and ciphertexts, respectively. Note that, $\mathcal{E}_K(x)$ and $\mathcal{D}_K(x)$ denote the results of encrypting/decrypting x with symmetric-key K. In addition, let $\mathcal{H}(\cdot)$ and $\mathcal{F}(\cdot)$ be hash functions that map $\langle g \rangle \to \mathcal{K}$ and $\langle g \rangle \to \mathbb{Z}/q\mathbb{Z}$, respectively.

At the end, for efficiency, we do NOT assume that all communications between players occur via a bulletin board. Considering this, however, we may need not only an authentication protocol, but also a non-repudiation protocol.

2.1 Overview

The main aim of this paper is to construct a Hybrid Mix scheme that provides routing flexibility without loss of computation/communication efficiency and security in terms of anonymity of the previous scheme as in [11].

As shown in Fig. 1, in the proposed scheme, each server, M_i, generates common key $K_{i,j}$, which is shared with user U_j using public/secret-key pair $G_{i-1,j}$ and a_i, respectively. Then, $K_{i,j}$ is used to decrypt $E_{i-1,j}$, and thereby obtains $E_{i,j}$ and Addr. Finally, the i-th server, M_i, outputs $(G_{i,j}, E_{i,j})$ to a server indicated by Addr that is the next server's address designated by U_j. This procedure is performed in the semi-asynchronous mode described in Sec. 1.3. Due to the address embedded in each ciphertext, the proposed Hybrid Mix scheme provides routing flexibility. However, note that the Hybrid Mix scheme proposed by Ohkubo and Abe provides no routing flexibility because each server's public key is sequentially generated using the previous server's public-key. In other words, the relay servers between message senders and the destination point are serially located and connected.

Here the proposed compressed key scheduling scheme is described briefly. The key sharing between M_1 and U_j is simply performed using the Diffie-Hellman key agreement, that is, $K_{1,j} = g^{r_j a_1} = G_{0,j}^{a_1} = h_1^{r_j}$. Note that, $G_{0,j}^{a_1}$ and $h_1^{r_j}$ are generated by M_1 using own secret a_1 and U_j using random number r_j, respectively. Then, in order to share the key between M_2 and U_j, the middle server, M_1, generates $G_{1,j} = G_{0,j}^{K_{1,j}}$ and sends it to M_2. This operation enables M_2 and U_j to generate common key $K_{2,j}$ by computing $G_{1,j}^{a_2}$ and $h_2^{r_j K_{1,j}}$, respectively. More generally, M_i and U_j obtain common key $K_{i,j}$ to compute $G_{i-1,j}^{a_i}$ and $h_i^{r_j K_{1,j} \cdots K_{i-1,j}}$, respectively, where $G_{i-1,j} = G_{i-2,j}^{K_{i-1,j}}$ and $G_{0,j} = g^{r_j}$. Note that, it is hard even for all relay servers except M_i to compute $K_{i,j}$ without secret a_i or r_j. Section 2.4 considers security in detail.

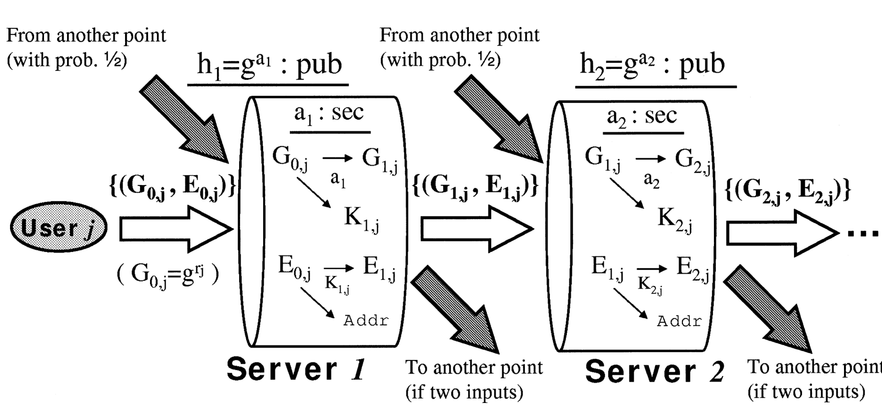

Fig. 1 Outline of the proposed Hybrid Mix scheme. Output ciphertexts from each server are individually sent to other servers indicated by Addr, which is the next server's address designated by each user.

2.2 Full Description

We now present the full construction of the proposed basic Hybrid Mix scheme. As a setup, all relay servers M_i ($1 \leq i \leq m$) *independently*† perform the pre-computation for a compressed key scheduling as follows:

[Setup]

Input: g.

1. Select private key a_i from $\mathbb{Z}/q\mathbb{Z}$ randomly.
2. Compute $h_i = g^{a_i}$ and publish h_i as a public-key of M_i.

Subsequently, U_j ($1 \leq j \leq n$) selects a number of relay servers and a routing, which is constituted by the servers, after he/she determines the destination point whose address is Addr^j_{dest}. We denote each i-th relay server selected by U_j and the address of the server as M_{j_i} and $\mathsf{Addr}^j_{j_i}$ ($1 \leq j_i \leq m$), respectively. Also, let us set j_i to i for simplicity.

Next, U_j ($1 \leq j \leq n$) performs a compressed key scheduling protocol to generate session keys $K_{i,j}$ ($i = 1, \ldots, m_j$) as follows:

[Compressed Key Scheduling by User]

Input: g, h_i $(i = 1, \ldots, m_j)$.

1. Compute $G_{0,j} = g^{r_j}$, where r_j is randomly taken from $\mathbb{Z}/q\mathbb{Z}$ and set $i \leftarrow 1$.
2. Compute $T_{i,j} = h_i^{r_j W_{i-1,j}}$ and $K_{i,j} = \mathcal{H}(T_{i,j})$, where $W_{i,j} = \mathcal{F}(T_{i,j}) \times W_{i-1,j}$ ($W_{0,j}$ is set to 1).
3. If $i < m_j$, then set $i \leftarrow i + 1$ and go to Step 2.

Then, U_j makes the following ciphertext embedding message $msg_j \in \mathcal{M}$, relay servers' addresses Addr^j_i ($i = 1, \ldots, m_j$) and destination point address Addr^j_{dest}. Note that, each address embedded in layers in a ciphertext only gives partial information of the entire route.

[Encryption]

Input: $K_{i,j}$, Addr^j_i $(i = 1, \ldots, m_j)$,
msg_j, Addr^j_{dest}.

1. Compute
$E_{m_j-1,j} = \mathcal{E}_{K_{m_j,j}}(RAND(\mathsf{Addr}^j_{dest} \| msg_j))$ and set $i \leftarrow m_j - 2$.
2. Compute
$E_{i,j} = \mathcal{E}_{K_{i+1,j}}(RAND(\mathsf{Addr}^j_{i+2} \| E_{i+1,j}))$.
3. If $i > 0$, then set $i \leftarrow i - 1$ and go to Step 2.

Here $RAND(\cdot)$ is denoted as an appropriate randomization function that maps to itself (i.e., $\mathcal{M}$). This function is required to extend the message selected by a user to the *uniformly random* data in the message space, $\mathcal{M}$. As a simple solution, the *cipher block chaining* (CBC) mode with a random initial value can be used. Note that, we need to adopt the random initial value only for the first one block (e.g. 128bits) message. In the CBC mode, the message is XORed with the previous ciphertext block before it is encrypted. In addition, to improve security, we claim such a randomization is needed for the scheme proposed by Ohkubo and Abe [11]††.

After performing the above procedures, U_j sends the pair of session-key encapsulation header and ciphertext, $C_{0,j} = (G_{0,j}, E_{0,j})$, to M_1.

On the other hand, relay server M_i ($1 \leq i \leq m$) who receives pairs of session-key encapsulation header and ciphertext, $C_{i-1,j} = (G_{i-1,j}, E_{i-1,j})$, performs the below compressed key scheduling protocol. As stated in Sec. 1.3, M_i basically outputs two pairs of ciphertext $C_{i,j}$ and shuffles the data after receiving two inputs $C_{i-1,j}$ from the previous relay servers.

[Compressed Key Scheduling by Server]

Input: $G_{i-1,j}$ $(\in C_{i-1,j})$.

1. Compute session keys $K_{i,j}$ for decryption algorithm $\mathcal{D}$ as
$$T_{i,j} = G^{a_i}_{i-1,j} \text{ and } K_{i,j} = \mathcal{H}(T_{i,j}).$$
2. Compute
$$G_{i,j} = G^{\mathcal{F}(T_{i,j})}_{i-1,j}.$$

Here $G_{i,j}$ are session-key encapsulation headers for the subsequent relay servers.

Finally, after M_i identifies the next relay servers by decrypting $E_{i-1,j}$ in $C_{i-1,j}$, the server outputs decryption results of the ciphertexts, $E_{i,j}$, and shuffles them. The specific procedure is as follows:

[Decryption and Shuffling]

Input: $G_{i,j}$, $E_{i-1,j}$, $K_{i,j}$.

1. Decrypt ciphertext $E_{i-1,j}$ as
$$\mathcal{D}_{K_{i,j}}(E_{i-1,j}) \rightarrow (\mathsf{Addr}^j_{i+1} \| E_{i,j}).$$
(Here, we omit the description for decoding the randomized message for simplicity. Note that, $\mathsf{Addr}^j_{i+1} = \mathsf{Addr}^j_{dest}$ when $i + 1 = m_j$.)
2. Respectively send $C_{i,j} = (G_{i,j}, E_{i,j})$ to Addr^j_{i+1} in a random order.

†In the key generation phase presented in [11], every server generates a public-key using the previous server's public-key as the key generation phase. This means that serially-located servers sequentially perform the key generation phase. Therefore, we can see that our key generation scheme is more efficient than the scheme presented in [11].

††More precisely, this operation is needed to discuss the security based on the Matching Find-Guess problem that is defined in [11] and stated in Sec. 2.4.2.

Table 2 Comparison of computational efficiency.

	trivial OR scheme	proposed scheme
Phase 1	mnC_{enc}	$(m+1)nC_{me}$
Phase 2	mnC_{dec}	$2mnC_{me}$

Table 3 Comparison of communication efficiency.

	trivial OR scheme	proposed scheme
Phase 1	$mnL_{pk} + nL_{msg}$	$nL_{pk} + nL_{msg}$
Phase 2	$\frac{m(m-1)}{2}nL_{pk} + mnL_{msg}$	$(m-1)nL_{pk} + mnL_{msg}$

We can see that message msg_j of U_j is finally sent to the sender's intended destination address, Addr^j_{dest}, if every M_i performs the procedure correctly.

2.3 Efficiency

Here, we quantitatively estimate the computation and communication efficiency of the proposed scheme. Tables 2 and 3 show the computational costs of the trivial Onion Routing (OR) scheme and the proposed scheme, and the communication costs of their schemes, respectively. In order to handle efficiently messages of various length, we suppose that the OR scheme adopts hybrid encryption employing both a public-key algorithm and symmetric-key encryption, in particular, RSA is used in the trivial OR scheme for key-delivery. Note that, to improve the security, it is desirable that we employ a probabilistic encryption in the trivial OR scheme. Terms C_{me}, C_{enc} and C_{dec} denote the computational complexity of a modular exponentiation in $(\mathbb{Z}/p\mathbb{Z})^*$ and of RSA encryption/decryption, respectively. Terms L_{msg} and L_{pk} denote the bit-length of a message and p (or RSA modulus N), respectively.

We assume that all communications between players occur via an authenticated channel. We ignore the size of the routing header for simplicity. Phase 1 means the encryption phase in case that every user employs m relay servers. Phase 2 means the decryption and shuffling phase in case that every relay server performs the procedure for n input ciphertexts. As shown in Table 2, the computational cost in the trivial OR scheme is slightly more efficient than that in the proposed scheme. However, as shown in Table 3, the communication cost in the trivial OR scheme is much more inefficient than that in the proposed scheme. Although the computational efficiency of the proposed scheme is worse than that of the OR scheme, we see that the proposed scheme achieves the optimal length of ciphertext since the communication efficiency level of each user and relay server are independent of the number of servers.

2.4 Security

This section addresses security in terms of anonymity of the proposed scheme. Roughly speaking, we say that a relay server provides anonymity if an adversary cannot distinguish the relation between two inputs to the relay server and two outputs from it. Obviously, the more there are such relay servers, the stronger anonymity that is provided by the proposed scheme is. In this section, we show that every incorruptible relay server, which operates according to the procedure described in Sec. 2.2 or Appendix B, provides anonymity in the presence of an adversary specified in Sec. 2.4.2.

2.4.1 Replay Attacks

Some of the existing anonymous networks may be vulnerable to replay attacks. The replay attack described in this paper is an attack on Mix-net to break its anonymity by re-sending messages to a relay server and comparing the outputs. For instance, we suppose that there is an adversary who has two messages that were input to a relay server and the output corresponding to either of the inputs. Then the adversary would clearly distinguish it by performing the replay attack on the relay server if the server outputs a unique result for any input. In fact, if any intermediate router in an anonymous network outputs a unique result for an input, an adversary in the network can trace the input by performing a replay attack. To prevent such attacks, Onion Routing [15] embeds an expiration time within each ciphertext. The approach is further strengthened by maintaining hash data of previously passed ciphertexts during a given time interval. In this paper, we assume that our scheme prevents any replay attack by using the techniques stated above, however, the procedure is omitted for the sake of simplicity.

2.4.2 Adversary Model

Subject to the considerations described in [1], the adversary model of Mix-net is defined from a combination of the following types:

active/passive: An *active* adversary can thoroughly control any colluding player, but a *passive* adversary has only the advantage of obtaining all the colluding players' secrets.

adaptive/static: An *adaptive* adversary can collude with appropriate players at an arbitrary time while a *static* adversary can only collude with them in advance.

2.4.3 Outline

For discussing user anonymity in the proposed scheme, we consider indistinguishability between two messages from an incorruptible relay server according to the previous study [11]. We first aim at an adversary who is just passive. In this case, an adversary aims to distinguish the relation between two incorruptible inputs to an incorruptible relay server and two outputs from it. We show that every incorruptible relay server, which operates according to the procedure described in Sec. 2.2, provides anonymity in the presence of a passive and adaptive adversary assuming the intractability of both *Gap Diffie-Hellman (GDH) Problem* and *Matching Find-Guess (MFG) Problem* in the random oracle model.

Next, we consider an adversary who is passive against servers, but active against users. In this case, by using *non-malleable* encryption technique, we can obtain a similar anonymity to the case that an adversary is just passive.

To prevent any attack by an adversary who is active against servers, we adopt the robust scheme using MAC [9] to the proposed basic scheme. Due to the combinational technique, our modified scheme provides anonymity unless the MAC based scheme is broken.

2.4.4 Definitions

The GDH and MFG problems introduced in the previous section are defined as follows:

Definition 1 (GDH Problem [12]) *We define the GDH problem as a computational problem of probabilistic polynomial-time adversary $\mathcal{A}_{GDH}$, who can make queries to a Decision Diffie-Hellman (DDH) Oracle, that plays the following game:*

Input: (p, q, g), (g^α, g^β), *where* $\alpha, \beta \in_R \mathbb{Z}/q\mathbb{Z}$.
Output: $g^{\alpha\beta}$.

If there is no help from the DDH oracle for $\mathcal{A}_{GDH}$, the above problem is identical to the *Computational Diffie-Hellman (CDH) Problem*. The CDH problem is a computational problem related to the Diffie-Hellman key-exchange [4]. On the other hand, the DDH oracle is an oracle that solves *the Decision Diffie-Hellman Problem* [8] with probability 1, that is, it answers whether or not $\gamma \in \mathbb{Z}/q\mathbb{Z}$ satisfies $\gamma = \alpha\beta$ when $\{(p, q, g), (g^\alpha, g^\beta, g^\gamma)\}$ is queried.

Definition 2 (MFG Problem [11]) *We define the MFG problem as a "matching" problem of probabilistic polynomial-time adversary $\mathcal{A}_{MFG}$ that plays the following game:*

Input: $(\mathcal{E}_{K_0}(msg_0), \mathcal{E}_{K_1}(msg_1), msg_b, msg_{\bar{b}})$, *where* $K_0, K_1 \in_R \mathcal{K}$, $msg_0, msg_1 \in_R \mathcal{M}$, $b \in_R \{0, 1\}$, *and* $\bar{b} = 1 - b$.
Output: b.

In addition, we formulate a new matching problem regarding anonymity of the proposed compressed key scheduling as stated in Sec. 2.2.

Definition 3 (Matching Compressed Key Scheduling (MCKS) Problem) *We define the MCKS problem as a matching problem of probabilistic polynomial-time adversary $\mathcal{A}_{MCKS}$ that plays the following game:*

Input: (p, q, g), $(g_0, g_1, g^a, g_b^{\mathcal{F}(g_{\bar{b}}^a)}, g_{\bar{b}}^{\mathcal{F}(g_b^a)})$, *where* $g_0, g_1 \in_R \langle g \rangle$, $a \in_R \mathbb{Z}/q\mathbb{Z}$, $b \in_R \{0, 1\}$, $\bar{b} = 1 - b$.
Output: b.

We discuss the security of the MCKS problem and obtain a positive result as shown by Lemma 5.

Finally, we introduce a new problem obtained by bundling two matching problems above.

Definition 4 (MFG-MCKS Joint Problem) *We define the MFG-MCKS joint problem as a matching problem of probabilistic polynomial-time adversary $\mathcal{A}_{MMJ}$ that plays the following game:*

Input: (p, q, g), $(g_0, g_1, g^a, g_b^{\mathcal{F}(g_{\bar{b}}^a)}, g_{\bar{b}}^{\mathcal{F}(g_b^a)}, \mathcal{E}_{K_0}(msg_0), \mathcal{E}_{K_1}(msg_1), msg_b, msg_{\bar{b}})$, *where* $g_0, g_1 \in_R \langle g \rangle$, $a \in_R \mathbb{Z}/q\mathbb{Z}$, $b \in_R \{0, 1\}$, $\bar{b} = 1 - b$, $K_0 = \mathcal{H}(g_0^a)$, $K_1 = \mathcal{H}(g_1^a)$, *and* $msg_0, msg_1 \in_R \mathcal{M}$.
Output: b.

2.4.5 Analysis

We first target an adversary who is just passive. Then we obtain the following lemmas.

Lemma 5 *We assume that $\mathcal{F}$ is a random oracle that maps $\langle g \rangle \to \mathbb{Z}/q\mathbb{Z}$. Then, if $\mathcal{A}_{MCKS}$ wins the game with non-negligible probability better than 1/2, $\mathcal{A}_{GDH}$ also wins the game with non-negligible probability with the help of $\mathcal{A}_{MCKS}$.*

The proof is given in Appendix A.1.

Lemma 6 *We assume that $\mathcal{H}$ and $\mathcal{F}$ are random oracles that map $\langle g \rangle \to \mathcal{K}$ and $\langle g \rangle \to \mathbb{Z}/q\mathbb{Z}$, respectively. Then, there exists an adversary that solves either the MFG problem or the MCKS problem with non-negligible probability better than 1/2 if $\mathcal{A}_{MMJ}$ wins the game with a similar probability.*

Lemma 6 can be proven in a straightforward manner by applying the proof technique for Lemma 2 in [11]. Roughly speaking, due to a random oracle model, we can consider that $\mathcal{H}(g_0^a)$ and $\mathcal{H}(g_1^a)$ are independent of the instance of the MCKS problem, $(g_0, g_1, g^a, g_b^{\mathcal{F}(g_{\bar{b}}^a)}, g_{\bar{b}}^{\mathcal{F}(g_b^a)})$, unless the CDH problem is solved; therefore instances of the MFG and MCKS problems can be separated even if $K_0 = \mathcal{H}(g_0^a)$ and $K_1 = \mathcal{H}(g_1^a)$. A similar technique is used in the proof of Lemma 5.

Lemma 7 *We assume that $\mathcal{H}$ and $\mathcal{F}$ are random oracles that map $\langle g \rangle \to \mathcal{K}$ and $\langle g \rangle \to \mathbb{Z}/q\mathbb{Z}$, respectively, and RAND is an ideal randomization function that maps to the message space in a uniformly random manner. We suppose that probabilistic polynomial-time adversary $\mathcal{A}$ is passive and adaptive. We also suppose that players M_k $(1 \leq k \leq m)$, U_0, and U_1 are incorruptible by $\mathcal{A}$ in our Mix-net and that U_0 and U_1 employ M_k concurrently. Then, $\mathcal{A}$ solves the MFG-MCKS joint problem with non-negligible probability better than 1/2 if he breaks the anonymity that M_k provides to U_0 and U_1.*

The proof is given in Appendix A.2.

Here, we reach the statement below stepwise using Lemmas 7, 6 and 5.

Theorem 8 *Our Mix-net provides anonymity in the presence of a probabilistic polynomial-time adversary who is characterized as passive and adaptive if multiple incorruptible users concurrently employ the incorruptible server, and both the GDH problem and the MFG problem are intractable.*

Next, we consider an adversary who is passive against servers, but active against users. In this case, as stated in [11], the underlying encryption must be made non-malleable to prevent any attack by users. If we assume that communication between a player and the corresponding first relay server occurs via a bulletin board, we can repel any attack by users by using a *proof of knowledge*. This means U_j proves to certain servers that it knows $r_j = \log_g G_{0,j}$ by using a zero-knowledge proof. However, the route between any user and the bulletin board is not ensured anonymity, hence the manager of the bulletin board may easily know who and when the Mix-net system is used. To prevent this problem, U_j proves to M_{j_i} for $1 \le i \le m_j$ that it knows $\log_{h_i} G_{i-1,j}$ by using a *non-interactive* zero-knowledge proof (NIZKP). Now, let $\mathsf{PoK}_{i,j}$ be a view of NIZKP for M_i generated by U_j. The view is for a proof of knowledge of $r_j \prod_{\ell=1}^{i-1} \mathcal{F}(T_{\ell,j})$ $(= \log_{h_i} G_{i-1,j})$. We see that U_j generates r_j and $T_{\ell,j}$, and thereby can make $\mathsf{PoK}_{i,j}$. Then, we need to replace ciphertext $E_{i,j} = \mathcal{E}_{K_{i+1},j}(RAND(\mathsf{Addr}^j_{i+2} \| E_{i+1,j}))$, which appears in the encryption phase to

$$E_{i,j} = \mathcal{E}_{K_{i+1},j}(RAND(\mathsf{PoK}_{i,j} \| \mathsf{Addr}^j_{i+2} \| E_{i+1,j})).$$

The latter revision may be rather inefficient because of the growth in the length of the ciphertext; therefore, it does not give the optimal length of ciphertext. However, the alternative revision provides the following statement:

Lemma 9 *We assume that $\mathcal{H}$ and $\mathcal{F}$ are random oracles that map $\langle g \rangle \to \mathcal{K}$ and $\langle g \rangle \to \mathbb{Z}/q\mathbb{Z}$, respectively, and RAND is an ideal randomization function that maps to the message space in a uniformly random manner. We suppose probabilistic polynomial-time adversary $\mathcal{A}$ is passive against servers, but active against users, and is adaptive. Furthermore, we suppose that players M_k $(1 \le k \le m)$, U_0, and U_1 are incorruptible by $\mathcal{A}$ in our Mix-net and that U_0 and U_1 employ M_k concurrently. Then, $\mathcal{A}$ solves the MFG-MCKS joint problem with non-negligible probability better than 1/2 if it breaks the anonymity that M_k provides to U_0 and U_1.*

Lemma 9 can be proven from Lemma 4 in [11] directly. Therefore, we reach the statement below stepwise using Lemmas 9, 6 and 5.

Theorem 10 *Our Mix-net provides anonymity in the presence of a probabilistic polynomial-time adversary who is characterized as passive against servers, active against users, and is adaptive if multiple incorruptible users concurrently employ the incorruptible server and both the GDH problem and the MFG problem are intractable.*

The security of the proposed scheme against an adversary who is active against servers must still be addressed. We discuss the way to prevent any attack by a server in Sec. 3.2.

3. Further Extensions

3.1 With Return Addresses

In this section, we extend the proposed basic scheme to a scheme that enables the return of a message from a receiver who received an anonymous message to the anonymous sender.

In the proposed scheme, basically, an anonymous sender selects both outward and homeward paths. Note that, these can be different paths.

Here, the proposed compressed key scheduling protocol to return a message is described briefly. For simplicity, we assume that a destination point that returns message $\widetilde{msg}_j$ to anonymous sender U_j is one of the relay servers. In addition, we omit the description for randomizing data and for decoding the randomized data as stated in the encryption phase and the decryption and shuffling phase in Sec. 2.2. The key generation phase and routing selection phase are essentially the same as that in Sec. 2.2. However, a sender also needs to select some relay servers that construct a homeward path and to obtain their addresses, $\widetilde{\mathsf{Addr}}^j_i$ $(i = 1, \ldots, \tilde{m}_j)$.

As stated in Sec. 2.2, the last relay server, M_{m_j} (i.e., the destination point for U_j), inputs $C_{m_j-1,j} = (G_{m_j-1,j}, E_{m_j-1,j})$ and then obtains message msg_j by generating $K_{m_j,j} = G^{a_{m_j}}_{m_j-1,j}$ and decrypting $E_{m_j-1,j}$. A feature of the extended scheme is to embed a homeward path in $E_{m_j-1,j}$. In this way, M_{m_j} obtains $(msg_j \| \widetilde{\mathsf{Addr}}^j_1 \| \tilde{E}_{0,j})$ by operation $\mathcal{D}_{K_{m_j,j}}(E_{m_j-1,j})$ and is able to send a triplet of session-key encapsulation header and two ciphertexts, $\tilde{C}_{0,j} = (\tilde{G}_{0,j}, \tilde{E}_{0,j}, \tilde{D}_{0,j})$, indicated by $\widetilde{\mathsf{Addr}}^j_1$, where $\tilde{G}_{0,j} = G^{\mathcal{F}(K_{m_j,j})}_{m_j-1,j}$ and $\tilde{D}_{0,j} = \mathcal{E}_{K_{m_j,j}}(\widetilde{msg}_j)$. Another feature of the extended scheme is that each relay server that is used in the construction of the homeward path further encrypts a received ciphertext of $\widetilde{msg}_j$. More precisely, one of the relay servers that is used in the construction of the homeward path for U_j, M_i, inputs $\tilde{C}_{i-1,j} = (\tilde{G}_{i-1,j}, \tilde{E}_{i-1,j}, \tilde{D}_{i-1,j})$ at first. After that, M_i generates common key, $\tilde{K}_{i,j} = \mathcal{H}(\tilde{G}^{a_i}_{i-1,j})$, using own secret a_i and obtains $(\widetilde{\mathsf{Addr}}^j_{i+1} \| \tilde{E}_{i,j})$ by operation $\mathcal{D}_{\tilde{K}_{i,j}}(\tilde{E}_{i-1,j})$. In addition, the relay server encrypts $\tilde{D}_{i-1,j}$ using $\tilde{K}_{i,j}$ and sends $\tilde{D}_{i,j} = \mathcal{E}_{\tilde{K}_{i,j}}(\tilde{D}_{i-1,j})$ to the next relay server indicated by $\widetilde{\mathsf{Addr}}^j_{i+1}$. Finally, anonymous sender U_j receives

$$\tilde{D}_{\tilde{m}_j,j} = \mathcal{E}_{\tilde{K}_{\tilde{m}_j,j}}(\cdots \mathcal{E}_{\tilde{K}_{1,j}}(\mathcal{E}_{K_{m_j,j}}(\widetilde{msg}_j)) \cdots),$$

and obtains $\widetilde{msg}_j$ by decrypting $\tilde{D}_{\tilde{m}_j,j}$ for $\tilde{m}_j + 1$ times. Note that, U_j can generate common keys $\tilde{K}_{i,j}$ for $1 \le i \le \tilde{m}_j$ in the same manner as stated in Sec. 2.2. Intuitively, $\tilde{K}_{i,j}$ can be represented by $h_i^{r_j K_{1,j} \cdots K_{m_j,j} \tilde{K}_{1,j} \cdots \tilde{K}_{i-1,j}}$.

3.2 Robust Scheme

This section presents a robust scheme to prevent any attack by servers using the proposed Hybrid Mix scheme. The specific protocols are given in the Appendix B. The protocols provide optimal robustness to our basic scheme.

The key idea for preventing any active attack by servers is to provide correctness in terms of each server's outcome by using a Message Authentication Code (MAC). This concept using a MAC is presented by Jakobsson and Juels in [9]. Using their technique, the corruption of any server can be detected by *honest* server managers who comprise more than half of all server managers. Details are given in [9].

Recently, Abe and Imai pointed out the weakness in terms of anonymity of the scheme presented in [9]. According to the remarks in [1], the weakness is removed if each user proves knowledge of the user's own secret for non-malleability. On the other hand, their scheme only provides server correctness with respect to a list that comprises ciphertexts from honest users even if the scheme is revised as stated above. This means that the ability of an adversary who is active against servers changes to being active against users but passive against servers. Therefore, since we already presented a way to prevent an active attack by any corrupt user, we can prevent any active attack by applying the technique in [9] and the repairs in [1].

We introduce the modified scheme based on their techniques in the Appendix B. In this paper, we do not discuss the security of the modified scheme, but we can refer to the security analysis in [9] and the revised version [1] because the concepts of our modified scheme are almost the same as the concept presented in [9].

4. Conclusion

We proposed an efficient Hybrid Mix scheme that provides both routing flexibility and the optimal length of ciphertext. We also extended the proposed basic scheme to a scheme that enables the return of a message from a receiver who received an anonymous message to the anonymous sender. These features enable the achievement of general and efficient communications for anonymous users in E-mail messages, Web services, and peer-to-peer applications. In addition, we showed that our basic scheme provides anonymity of the message sender if a realistic adversary is restricted to being passive and then provides the optimal length of ciphertext. We also showed that our scheme can prevent any active attack by incorporating and improving existing efficient approaches. Therefore, our scheme can construct secure anonymous communications and can be applied to various applications.

References

[1] M. Abe and H. Imai, "Flaws in some robust optimistic Mix-nets," Eighth Australasian Conference on Information Security and Privacy (ACISP '03), LNCS 2727, pp.39–51, Springer-Verlag, 2003.

[2] D.L. Chaum, "Untraceable electronic mail, return address, and digital pseudonyms," Commun. ACM, vol.24, no.2, pp.84–88, 1981.

[3] Y. Desmedt and K. Kurosawa, "How to break a practical MIX and design a new one," Eurocrypt '00, LNCS 1807, pp.557–572, Springer-Verlag, 2000.

[4] W. Diffie and M.E. Hellman, "New directions in cryptography," IEEE Trans. Inf. Theory, vol.22, no.6, pp.644–654, 1976.

[5] P. Feldman, "A practical scheme for non-interactive verifiable secret sharing," Proc. 28th IEEE Symposium on the Foundations of Computer Science (FOCS), pp.427–437, IEEE Press, 1987.

[6] Freedom, http://www.freedom.net/products/websecure/

[7] The free haven project, http://www.freehaven.net/

[8] H. Handschuh, Y. Tsiounis, and M. Yung, "Decision oracles are equivalent to matching oracles," Public Key Cryptography (PKC) '99, LNCS 1560, pp.276–289, Springer-Verlag, 1999.

[9] M. Jakobsson and A. Juels, "An optimally robust hybrid mix network," Proc. Twentieth ACM Symposium on Principles of Distributed Computing (PODC '01), pp.284–292, ACM Press, 2001.

[10] Mixmaster, http://sourceforge.net/projects/mixmaster/

[11] M. Ohkubo and M. Abe, "A length-invariant hybrid mix," ASIACRYPT '00, LNCS 1976, pp.178–191, Springer-Verlag, 2000.

[12] T. Okamoto and D. Pointcheval, "The gap-problems: A new class of problems for the security of cryptographic schemes," Public Key Cryptography (PKC) '01, LNCS 1992, pp.104–118, Springer-Verlag, 2001.

[13] C. Park, K. Itoh, and K. Kurosawa, "Efficient anonymous channel and all/nothing election scheme," EUROCRYPT '93, LNCS 765, pp.248–259, Springer-Verlag, 1994.

[14] K. Sako and J. Kilian, "Receipt-free mix-type voting scheme—A practical solution to the implementation of a voting booth," EUROCRYPT '95, LNCS 921, pp.393–403, Springer-Verlag, 1995.

[15] P.F. Syverson, D.M. Goldschlag, and M.G. Reed, "Anonymous connections and onion routing," Proc. 1997 IEEE Symposium on Security and Privacy, pp.44–54, IEEE Press, 1997.

Appendix A: Proofs

A.1 Proof of Lemma 5

After $\mathcal{A}_{GDH}$ receives an instance of the GDH problem, $\{(p,q,g),(g^{\alpha},g^{\beta})\}$, he makes $\mathsf{INS}_R^{\mathrm{GDH}} \stackrel{\mathrm{def}}{=} \{(p,q,g),((g^{\beta})^{t_0}, (g^{\beta})^{t_1}, g^{\alpha}, R_{b'}, R_{\bar{b}'})\}$ as an instance of the MCKS problem, $\mathsf{INS}_D^{\mathrm{GDH}} \stackrel{\mathrm{def}}{=} \{(p,q,g),(g_0,g_1,g^{a}, g_b^{\mathcal{F}(g_b^{\alpha})}, g_{\bar{b}}^{\mathcal{F}(g_{\bar{b}}^{\alpha})})\}$. Here $R_{b'}, R_{\bar{b}'} \in_R \langle g\rangle$ and $t_0, t_1 \in_R \mathbb{Z}/q\mathbb{Z}$. $b' \in_R \{0,1\}$ and $\bar{b}' \stackrel{\mathrm{def}}{=} 1-b'$ are obtained by flipping a coin. Then when assuming that random oracle $\mathcal{F}$ is assigned as

$$(R_{b'}, R_{\bar{b}'}) = \begin{cases} (\tilde{g}_0^{\mathcal{F}(\tilde{g}_0^{\alpha})}, \tilde{g}_1^{\mathcal{F}(\tilde{g}_1^{\alpha})}) & \text{if} \quad b'=0, \\ (\tilde{g}_1^{\mathcal{F}(\tilde{g}_1^{\alpha})}, \tilde{g}_0^{\mathcal{F}(\tilde{g}_0^{\alpha})}) & \text{otherwise}, \end{cases}$$

where $\tilde{g}_0 \stackrel{\mathrm{def}}{=} (g^{\beta})^{t_0}$ and $\tilde{g}_1 \stackrel{\mathrm{def}}{=} (g^{\beta})^{t_1}$, we see that $\mathsf{INS}_R^{\mathrm{GDH}} \in \{\mathsf{INS}_D^{\mathrm{GDH}}\}$. Note that, if $\mathcal{A}_{MCKS}$ makes a query, either $\tilde{g}_0^{\alpha}$ or $\tilde{g}_1^{\alpha}$, to $\mathcal{F}$, he may be able to receive no response from $\mathcal{F}$ because if this is the case, $\mathcal{F}$ must compute either $\log_{\tilde{g}_0} \tilde{g}_0^{\mathcal{F}(\tilde{g}_0^{\alpha})}$ or $\log_{\tilde{g}_1} \tilde{g}_1^{\mathcal{F}(\tilde{g}_1^{\alpha})}$ and this means that $\mathcal{F}$ solves the discrete logarithm problem for $(p,q,\tilde{g}_0,R_0)$ or $(p,q,\tilde{g}_1,R_1)$. On the other hand, if either $\tilde{g}_0^{\alpha}$ or $\tilde{g}_1^{\alpha}$ exists among queries Q_i for $1 \le i \le \mathrm{poly}(\rho)$ (where $\mathrm{poly}(\rho)$ is a polynomial function of security parameter ρ), $\mathcal{A}_{GDH}$ will win the game by performing the following procedure:

1. Input $\mathsf{INS}_R^{\mathrm{GDH}}$ to $\mathcal{A}_{MCKS}$.
2. For $1 \le i \le \mathrm{poly}(\rho)$, receive query Q_i which is input from $\mathcal{A}_{MCKS}$ to $\mathcal{F}$.
3. Generate two instances of the DDH problem, $\mathsf{INS}_{i,0}^{\mathrm{DDH}} \stackrel{\mathrm{def}}{=} \{(p,q,g),(g^\alpha,\tilde{g}_0,Q_i)\}$ and $\mathsf{INS}_{i,1}^{\mathrm{DDH}} \stackrel{\mathrm{def}}{=} \{(p,q,g),(g^\alpha,\tilde{g}_1,Q_i)\}$, and input them to the DDH oracle.
4. If the DDH oracle returns "Yes" for $\mathsf{INS}_{i_d,j}^{\mathrm{DDH}}$ ($j \in \{0,1\}$), then go to the next step. Otherwise, go back to Step 2.
5. Compute $Q_{i_d}^{1/t_j}$ and output it as the answer of the GDH problem.

Since $Q_{i_d}^{1/t_j} = (\tilde{g}_j^\alpha)^{1/t_j} = g^{\alpha\beta}$ holds, we see that $\mathcal{A}_{GDH}$ wins the game if $\tilde{g}_j^\alpha$ is included in $\{Q_i\}$ with non-negligible probability.

Next, we show that $\tilde{g}_j^\alpha$ is included in $\{Q_i\}$ with non-negligible probability. We denote by $\mathsf{E}_{\mathsf{Yes}}^{\mathcal{F}}$ an event that $\mathcal{A}_{MCKS}$ makes a query, $\mathsf{INS}_{i,j}^{\mathrm{DDH}}$, such that the DDH oracle returns "Yes". Then the probability that $\mathcal{A}_{MCKS}$ wins the game, $\mathsf{Win}_{\mathcal{A}_{MCKS}}^{\mathcal{F}}$, can be estimated as

$$\begin{aligned}\mathsf{Win}_{\mathcal{A}_{MCKS}}^{\mathcal{F}} = {} & \Pr[\mathsf{E}_{\mathsf{Yes}}^{\mathcal{F}}]\Pr[\mathcal{A}_{MCKS}\text{ wins } \mid \mathsf{E}_{\mathsf{Yes}}^{\mathcal{F}}] \\ & +\Pr[\neg\mathsf{E}_{\mathsf{Yes}}^{\mathcal{F}}]\Pr[\mathcal{A}_{MCKS}\text{ wins } \mid \neg\mathsf{E}_{\mathsf{Yes}}^{\mathcal{F}}]. \qquad (\mathrm{A}{\cdot}\,1)\end{aligned}$$

Based on the assumption of this lemma, $\mathsf{Win}_{\mathcal{A}_{MCKS}}^{\mathcal{F}}$ can also be estimated as $\frac{1}{2}+\theta$, where θ is a non-negligible function of the security parameter, ρ. On the other hand, $\Pr[\mathcal{A}_{MCKS}\text{ wins } \mid \neg\mathsf{E}_{\mathsf{Yes}}^{\mathcal{F}}]$ is $\frac{1}{2}$ because $R_{b'}$ and $R_{\bar{b}'}$ are independent of b' when $\mathsf{E}_{\mathsf{Yes}}^{\mathcal{F}}$ does not occur. Therefore, we can transform Eq. (A· 1) into

$$2\theta = \Pr[\mathsf{E}_{\mathsf{Yes}}^{\mathcal{F}}]\,(2\Pr[\mathcal{A}_{MCKS}\text{ wins } \mid \mathsf{E}_{\mathsf{Yes}}^{\mathcal{F}}]-1),$$

and this formula represents that $\Pr[\mathsf{E}_{\mathsf{Yes}}^{\mathcal{F}}]$ is non-negligible unless $\Pr[\mathcal{A}_{MCKS}\text{ wins } \mid \mathsf{E}_{\mathsf{Yes}}^{\mathcal{F}}]$ equals $\frac{1}{2}$. However, if $\Pr[\mathcal{A}_{MCKS}\text{ wins } \mid \mathsf{E}_{\mathsf{Yes}}^{\mathcal{F}}]$ equals $\frac{1}{2}$, $\mathsf{Win}_{\mathcal{A}_{MCKS}}^{\mathcal{F}}$ also becomes $\frac{1}{2}$; this is inconsistent with the assumption of this lemma.

A.2 Proof of Lemma 7

We denote a simulator that breaks the anonymity that M_k provides to U_0 and U_1 by SIM. From the protocol stated in Sec. 2.2, we can set an input to SIM as

$$\begin{aligned}\mathsf{L}_D \stackrel{def}{=} {} & ((p,q,g),G_{k-1,0},G_{k-1,1},h_k,G_{k,b},G_{k,\bar{b}}, \\ & E_{k-1,0},E_{k-1,1},E_{k,b},E_{k,\bar{b}}),\end{aligned}$$

where $(p,q,g) \leftarrow \mathcal{G}(1^\rho)$ (ρ: security parameter), $G_{i,\ell} = G_{i-1,\ell}^{\mathcal{F}(T_{i,\ell})}$, $T_{i,\ell} = G_{i-1,\ell}^{a_i}$, $G_{0,\ell} = g^{r_\ell}$, $r_\ell \in_R \mathbb{Z}/q\mathbb{Z}$ ($\ell = 1,2$), $h_k = g^{a_k}$, $a_k \in_R \mathbb{Z}/q\mathbb{Z}$, $b \in_R \{0,1\}$, $\bar{b} = 1-b$, $E_{i,\ell} = \mathcal{E}_{K_{i+1,\ell}}(E_{i+1,\ell})$ and $K_{i,\ell} = \mathcal{H}(G_{i-1,\ell}^{a_i})$. We assume without loss of generality that each address, $\mathsf{Addr}_{k+1}^{\ell}$, which is designated by U_ℓ as the address of the next relay server of M_k, is included in $E_{k,\ell}$. In addition, $E_{k,\ell}$ are assumed to be independently random in $\mathcal{M}$ because of the appropriate randomization scheme as stated in the body of the paper.

Let $\mathcal{A}$ be a probabilistic polynomial-time adversary who tries to break the MFG-MCKS joint problem by using SIM. Then, $\mathcal{A}$ receives an instance of the MFG-MCKS joint problem,

$$\begin{aligned}\mathsf{L}_R \stackrel{def}{=} {} & ((p,q,g),g_0,g_1,g^a,g_b^{\mathcal{F}(g_b^a)},g_{\bar{b}}^{\mathcal{F}(g_{\bar{b}}^a)},\mathcal{E}_{K_0}(msg_0), \\ & \mathcal{E}_{K_1}(msg_1),msg_b,msg_{\bar{b}}),\end{aligned}$$

where $g_0,g_1 \in_R \langle g\rangle$, $a \in_R \mathbb{Z}/q\mathbb{Z}$, $K_0 = \mathcal{H}(g_0^a)$, $K_1 = \mathcal{H}(g_1^a)$ and $msg_0,msg_1 \in_R \mathcal{M}$. Now, if L_D and L_R are indistinguishable for SIM, we can see that $\mathcal{A}$ wins the game. So hereafter, we consider this indistinguishability.

At first, we can clearly see that $((p,q,g),h_k,E_{k,b},E_{k,\bar{b}})$ in L_D and $((p,q,g),g^a,msg_b,msg_{\bar{b}})$ in L_R are perfectly indistinguishable. Next, we consider the indistinguishability between $(G_{k-1,0},G_{k-1,1})$ in L_D and (g_0,g_1) in L_R. It is defined that the latter pair is independently random. On the other hand, the former pair is also assumed independently random. We sketch the proof. If $k=1$, then $G_{k-1,0} = G_{0,0} = g^{r_0}$ and $G_{k-1,1} = G_{0,1} = g^{r_1}$ hold and are independently random because $r_0,r_1 \in_R \mathbb{Z}/q\mathbb{Z}$. If $k>1$, $G_{k-1,\ell}$ is represented by $G_{k-2,\ell}^{\mathcal{F}(T_{k-1,\ell})}$, where $T_{k-1,\ell} = G_{k-2,\ell}^{a_{k-1}}$ and $a_{k-1} \in_R \mathbb{Z}/q\mathbb{Z}$. Especially, $G_{1,\ell} = g^{r_\ell\mathcal{F}(T_{1,\ell})}$, where $T_{1,\ell} = g^{r_\ell a_1}$ holds. This representation directly leads to a result that $G_{1,0}$ and $G_{1,1}$ are independently random except for a negligible error probability in the case of $r_0 = r_1$. Similarly, $G_{k-1,0} = g^{r_0\mathcal{F}(T_{1,0})\cdots\mathcal{F}(T_{k-1,0})}$ and $G_{k-1,1} = g^{r_1\mathcal{F}(T_{1,1})\cdots\mathcal{F}(T_{k-1,1})}$ are assumed to be independently random except for a negligible error probability in the case of $r_0 = r_1$. Therefore, we obtain the outcome that $(G_{k-1,0},G_{k-1,1})$ and (g_0,g_1) are indistinguishable. Then, from this outcome, we directly obtain the other outcome that two pairs $(G_{k,b},G_{k,\bar{b}}) = (G_{k-1,b}^{\mathcal{F}(G_{k-1,b}^{a_k})},G_{k-1,\bar{b}}^{\mathcal{F}(G_{k-1,\bar{b}}^{a_k})})$ and $(g_b^{\mathcal{F}(g_b^a)},g_{\bar{b}}^{\mathcal{F}(g_{\bar{b}}^a)})$ are indistinguishable.

At the end, we consider the indistinguishability between two pairs $(E_{k-1,0},E_{k-1,1}) = (\mathcal{E}_{K_{k,0}}(E_{k,0}),\mathcal{E}_{K_{k,1}}(E_{k,1}))$ and $(\mathcal{E}_{K_0}(msg_0),\mathcal{E}_{K_1}(msg_1))$. We can clearly see that they are indistinguishable if $K_{k,0}$, $K_{k,1}$, K_0 and K_1 are all taken as uniformly random from $\mathcal{K}$. On the other hand, $K_{k,\ell}$ and K_ℓ are represented by $\mathcal{H}(G_{k-1,\ell}^{a_k})$ and $\mathcal{H}(g_\ell^a)$. This means that $K_{k,\ell}$ and K_ℓ are uniformly random because $G_{k-1,0},G_{k-1,1},g_0,g_1 \in_R \langle g\rangle$, $a,a_k \in_R \mathbb{Z}/q\mathbb{Z}$ and $\mathcal{H}$ is a random oracle. Therefore, $(E_{k-1,0},E_{k-1,1})$ and $(\mathcal{E}_{K_0}(msg_0),\mathcal{E}_{K_1}(msg_1))$ are indistinguishable.

Finally, we reach the statement that L_D and L_R are indistinguishable for SIM; thus, proving this lemma.

Appendix B: Procedures of Robust Scheme

This section presents a Hybrid Mix scheme for preventing any attack by servers. This is based on the proposed basic Hybrid Mix scheme stated in Sec. 2.2. For simplicity, in this section we assume that all communications between players occur via a bulletin board. In addition, we omit the description for randomizing data and for decoding the randomized data as stated in the encryption phase and the decryption and shuffling phase in Sec. 2.2.

As a setup, all relay servers M_i ($1 \le i \le m$) independently perform the pre-computation for a compressed key scheduling as follows:

[Setup]

Input: g.

1. Select two private keys, a_i and d_i, from $\mathbb{Z}/q\mathbb{Z}$ randomly.
2. Compute $h_i = g^{a_i}$ and $z_i = g^{d_i}$, and publish them.
3. Share secrets a_i and d_i with other servers using t out of m Verifiable Secret Sharing, (t, m)-VSS [5].

We set $t = \lceil (m+1)/2 \rceil$ for optimal robustness.

After that, each user, U_j, selects a number of relay servers and a routing in a similar manner as stated previously, but we omit the procedure for simplicity.

Next, U_j ($1 \le j \le n$) performs a compressed key scheduling protocol to generate session keys $K_{i,j}$ and MAC keys $Z_{i,j}$ ($i = 1, \ldots, m_j$) as follows:

[Compressed Key Scheduling by User]

Input: $g,\ h_i,\ z_i \quad (i = 1, \ldots, m_j)$.

1. Compute $G_{0,j} = g^{r_j}$, where r_j is randomly taken from $\mathbb{Z}/q\mathbb{Z}$ and set $i \leftarrow 1$.
2. Compute $T_{i,j} = h_i^{r_j W_{i-1,j}}$, $K_{i,j} = \mathcal{H}(T_{i,j})$ and $Z_{i,j} = z_i^{r_j W_{i-1,j}}$, where $W_{i,j} = \mathcal{F}(T_{i,j}) \times W_{i-1,j}$ ($W_{0,j}$ is set to 1).
3. If $i < m_j$, then set $i \leftarrow i+1$ and go to Step 2.

Then, U_j makes the following ciphertext embedding message $msg_j \in \mathcal{M}$, relay servers' address Addr_i^j ($i = 1, \ldots, m_j$), destination point's address Addr_{dest}^j, MACs $\mu_{i,j}$ ($i = 1, \ldots, m_j - 1$) and MAC TS_j for authentication of the last stage, where TS_j is a MAC that can only be generated by U_j or honest server managers that exceed $m/2$. See [9] for details as we omit the procedure for simplicity. It is, however, that TS_j may be able to be checked by the last server by itself if the last server needs a valid message (e.g. the server that is a destination point and carries on an anonymous service site).

[Encryption]

Input: $K_{i,j}$, $Z_{i,j}$, Addr_i^j ($i = 1, \ldots, m_j$), msg_j, Addr_{dest}^j, SID.

1. Compute $E_{m_j-1,j} = \mathcal{E}_{K_{m_j,j}}(\mathsf{Addr}_{dest}^j \| msg_j \| \mathsf{TS}_j)$ and set $i \leftarrow m_j - 2$,
2. Compute $E_{i,j} = \mathcal{E}_{K_{i+1,j}}(\mathsf{Addr}_{i+2}^j \| E_{i+1,j} \| \mu_{i+1,j})$, where $\mu_{i+1,j} = MAC_{Z_{i+2,j}}[E_{i+1,j} \| \mathsf{SID}]$ for $i \ge 1$ and SID is a session ID. Note that, we denote by $MAC_Z[msg]$ the MAC under key Z of message msg.
3. If $i > 0$, then set $i \leftarrow i - 1$ and go to Step 2.

After performing the above procedures, U_j sends the pair of session-key encapsulation header and ciphertext, $C_{0,j} = (G_{0,j},\ E_{0,j})$, with PoK_j to M_1, where PoK_j is a proof of knowledge of $r = \log_g G_{0,j}$. It is assumed that PoK_j is published on a bulletin board.

On the other hand, relay server M_i ($1 \le i \le m$) who receives pairs of session-key encapsulation header and ciphertext, $C_{i-1,j} = (G_{i-1,j},\ E_{i-1,j})$, performs the below compressed key scheduling protocol.

[Compressed Key Scheduling by Server]

Input: $C_{i-1,j}$, $\mu_{i-1,j}$ SID.

1. Compute MAC keys as
$$\tilde{Z}_{i,j} = G_{i-1,j}^{d_i}.$$
2. Compute MACs as
$$\tilde{\mu}_{i-1,j} = MAC_{\tilde{Z}_{i,j}}[E_{i-1,j} \| \mathsf{SID}].$$
3. If $\mu_{i-1,j} = \tilde{\mu}_{i-1,j}$ do not hold, then go to the *verify complaint* protocol, which is stated at the last of this section.
4. Compute session keys $K_{i,j}$ for decryption algorithm $\mathcal{D}$ as
$$T_{i,j} = G_{i-1,j}^{a_i} \text{ and } K_{i,j} = \mathcal{H}(T_{i,j}).$$
5. Compute
$$G_{i,j} = G_{i-1,j}^{\mathcal{F}(T_{i,j})}.$$

Finally, after M_i identifies the next relay servers by decrypting $E_{i-1,j}$ in $C_{i-1,j}$, the server outputs decryption results of the ciphertexts, $E_{i,j}$, with shuffling them and the proof. The specific procedure is as follows:

[Decryption and Shuffling]

Input: $G_{i,j}$, $E_{i-1,j}$, $K_{i,j}$, $T_{i,j}$.

1. Decrypt ciphertext $E_{i-1,j}$ as
$$\mathcal{D}_{K_{i,j}}(E_{i-1,j}) \rightarrow (\mathsf{Addr}_{i+1}^j \| E_{i,j} \| \mu_{i,j}).$$
(Note that, $\mathsf{Addr}_{i+1}^j = \mathsf{Addr}_{dest}^j$ when $i = m_j$.)
2. Respectively send
$C_{i,j} = (G_{i,j},\ E_{i,j},\ \mu_{i,j})$ to Addr_{i+1}^j in a random order.
3. Prove knowledge of $\{\mathcal{F}(T_{i,j})\}$ in the equation
$\prod_j G_{i,j} = \prod_j G_{i-1,j}^{\mathcal{F}(T_{i,j})}$ using NIZKP.

Note that, the approach of zero-knowledge proof in Step 3 is different from that of [9]. This substitution slightly worsens as compared to the original scheme.

After that, honest managers of servers, the number of which exceed $m/2$, check whether $\mu_{m_j,j} \stackrel{def}{=} \mathsf{TS}_j$ is valid as the last validation check. If invalid, a verify complaint protocol is needed as described below.

In the case that some inconsistencies occur in the decryption phase of M_i or the last validation check, a verification phase is performed through cooperation of honest managers of servers, the number of which exceed $m/2$, as follows:

[Verify Complaint]

Input: $\tilde{j}$ (Here we assume that M_i complains about its own input $C_{i-1,\tilde{j}} = (G_{i-1,\tilde{j}}, E_{i-1,\tilde{j}}, \mu_{i-1,\tilde{j}})$ or $C_{0,\tilde{j}} = (G_{0,\tilde{j}}, E_{0,\tilde{j}}, \mathsf{PoK}_{\tilde{j}})$ sent from M_{i-1} or $\mathsf{U}_{\tilde{j}}$, respectively).

1. If M_i is the first relay server for $\mathsf{U}_{\tilde{j}}$ (i.e. $i = 1$), go to Step 3. Otherwise, perform the following procedure.
   a. Compute $\tilde{Z}_{i,\tilde{j}} = G_{i-1,\tilde{j}}^{d_i}$ without disclosing d_i using some shares of d_i.
   b. Check whether $\mu_{i-1,\tilde{j}} = MAC_{\tilde{Z}_{i,\tilde{j}}}[E_{i-1,j}||\mathsf{SID}]$ holds or not, and verify NIZKP of $\{\mathcal{F}(T_{i,j})\}$ in the equation $\prod_j G_{i,j} = \prod_j G_{i-1,j}^{\mathcal{F}(T_{i,j})}$. If both are valid, then go to Step 4.
2. Perform the following procedure after M_{i-1} publishes the input that corresponds to $C_{i-1,\tilde{j}}$.
   a. Compute $(G_{i-1,\tilde{j}}, E_{i-1,\tilde{j}})$ from the input $(G_{i-2,\tilde{j}}, E_{i-2,\tilde{j}})$ in $C_{i-2,\tilde{j}}$. If some inconsistencies occur, then go to Step 5.
   b. Compute $\tilde{Z}_{i-1,\tilde{j}} = G_{i-2,\tilde{j}}^{d_{i-1}}$ without disclosing d_{i-1} using some shares of d_{i-1}.
   c. Check whether $\mu_{i-2,\tilde{j}} = MAC_{\tilde{Z}_{i-1,\tilde{j}}}[E_{i-2,\tilde{j}}||\mathsf{SID}]$ holds or not. If invalid, then go to Step 5.
   d. Determine $C_{i-1,\tilde{j}}$ as a corrupted input, and go to Step 1 after setting $i \leftarrow i - 1$.
3. Verify $\mathsf{PoK}_{\tilde{j}}$. If invalid, then expel $C_{0,\tilde{j}}$ and go back to the procedure of Compressed Key Scheduling by server M_i. Otherwise, go to Step 4.
4. Expel M_i and perform the procedure of Compressed Key Scheduling by server M_i again.
5. Expel M_{i-1} and perform the procedure of Compressed Key Scheduling by server M_{i-1} again. Also, if possible, go back to the procedure of Compressed Key Scheduling by server M_i.

Koji Chida received his B.S. and M.S. degrees from Waseda University, Tokyo, Japan, in 1998 and 2000, respectively. He is presently engaged in research on Cryptography and Information Security at NTT Information Sharing Platform Laboratories. He is a member of the Information Processing Society of Japan. In 2001, he was awarded the SCIS 2000 paper prize.

Masayuki Abe received his B.S., and M.S. degrees from Tokyo University of Science in 1990 and 1992, respectively, and Ph.D. degree from the University of Tokyo in 2002. He is a senior research scientist of the NTT Information Sharing Platform Laboratories. Member of IACR.