Cryptanalysis and Improvement of a Multi-receiver Identity-Based Key Encapsulation at INDOCRYPT 06*

Jong Hwan Park
Center for Information Security Technologies
Korea University
decartian@korea.ac.kr

Ki Tak Kim
Center for Information Security Technologies
Korea University
kitak@korea.ac.kr

Dong Hoon Lee
Center for Information Security Technologies
Korea University
donghlee@korea.ac.kr

ABSTRACT

Multi-receiver Identity-Based Key Encapsulation Mechanism (mIB-KEM) allows a sender to distribute messages for a set of receivers using the receiver's identity as a public key. Recently, Chatterjee and Sarkar [12] suggested a new mIB-KEM which has sublinear-size ciphertexts and private keys simultaneously. They demonstrated that their scheme is secure against chosen plaintext (or ciphertext) attacks without random oracles. In this paper, we show that their scheme is not secure in that a revoked user can easily decrypt ciphertexts. We next propose a new mIB-KEM which overcomes the security flaw identified in the construction of Chatterjee and Sarkar.

Categories and Subject Descriptors

E.3 [**Data Encryption**]: Code breaking, Public key cryptosystems

General Terms

Security

Keywords

Multi-receiver Identity-Based Key Encapsulation

1. INTRODUCTION

To distribute a message to some set S of users, the trivial method is to encrypt the message using each user's public key. As expected, ciphertext size in this setting increases linearly with $|\mathsf{S}|$. This results in too much ciphertext size when S becomes a large set of receivers. Thus, this method is less attractive, except for the case where S is small. An alternative method is to encrypt a message encryption key which is used to encrypt the message under a symmetric key cipher. This is called a hybrid (KEM-DEM) encryption paradigm. This is efficient, in particular, for such applications as the sender wants to broadcast a same message. In this paper, we mainly focus on the hybrid paradigm.

In an identity-based setting [19], a sender is able to distribute a message for a set S of receivers using the receiver's identity as a public key. As usual, a Key Generation Center (KGC) issues a private key for a user identity, and public parameters for the identity-based system are shared with all users. Regarding the KEM-DEM encryption in multi-receiver setting, we can easily consider a trivial solution by concatenating ciphertexts for all receivers. One may attempt to do this with the previous Identity-Based Encryption (IBE) schemes [6, 13, 4, 5, 20, 16], where the Hierarchical IBE (HIBE) schemes [4, 5] are considered as 1-level IBE schemes. However, the IBE schemes [4, 5, 16] are not suitable for the simple solution.

For example, consider the Gentry IBE scheme [16] for a multi-receiver Identity-Based Key Encapsulation Mechanism (mIB-KEM). Assume a sender intends to broadcast a message to a set $\mathsf{S} = \{\mathsf{ID}_1, \ldots, \mathsf{ID}_k\}$. With elements g, h, and g_1 in the public parameters, ciphertext for the set S will be of the form $(g_1^s g^{-s\cdot\mathsf{ID}_1}, \ldots, g_1^s g^{-s\cdot\mathsf{ID}_k}, e(g,g)^s)$ for same randomness s, and then the message encryption key corresponding to this ciphertext will become $e(g,h)^{-s}$. Since information about S is broadcast together with the ciphertext, a revoked user with identity ID' (i.e., outside of S) can compute g^{-s} as $(g_1^s g^{-s\cdot\mathsf{ID}_1}/g_1^s g^{-s\cdot\mathsf{ID}_k})^{1/(\mathsf{ID}_1-\mathsf{ID}_k)}$ and obtain g_1^s. Thus, he can reconstruct $g_1^s g^{-s\cdot\mathsf{ID}'}$ and decrypt the ciphertext successfully. This observation can be applied to other mIB-KEMs based on the IBE schemes [4, 5] in the similar manner.

In this paper we show that a mIB-KEM suggested by Chatterjee and Sarkar [12] is not secure. We will show how a revoked user can easily decrypt ciphertext in [12]. Our security analysis is similar to that mentioned immediately above. Next, we suggest an improvement that overcomes the security flaw identified in [12]. In our construction, we partition an identity space into subsets using two publicly computable surjective functions, and we add to public parameters random elements which are representative of the subsets. These random values play the role of preventing such an attack above from occurring. The proposed mIB-KEM is secure against chosen plaintext attacks in the selective-ID model, and is extended to achieve chosen ciphertext security by using a hash-based method [9]. With appropriate parametriza-

*This research was supported by the MIC(Ministry of Information and Communication), Korea, under the ITRC(Information Technology Research Center) support program supervised by the IITA(Institute of Information Technology Advancement) (IITA-2007-(C1090-0701-0025)).

ASIACCS '08, March 18-20, Tokyo, Japan
Copyright 2008 ACM 978-1-59593-979-1/08/0003 ...$5.00.

tion, our scheme could have sublinear-size ciphertexts and private keys. We prove the security of our scheme without random oracles under the decision Bilinear Diffie-Hellman Exponent (BDHE) assumption, which was already used to prove security in [5, 7].

Related Works. Since the first practical constructions of the IBE primitives [13, 6] appeared, many research has been done to create secure mIB-KEMs [14, 2, 3, 11, 12, 1, 18]. Until now, all of the proposed mIB-KEMs made use of efficiently computable bilinear maps (i.e., pairing) on elliptic curve.

The first mIB-KEM [14] was suggested by Du et al. [14], based on the Boneh-Franklin IBE scheme [6]. Their construction obtained $O(|\mathsf{S}|)$ ciphertexts and $O(1)$ private keys for a receiver set S, but formal security proof was not provided. Later, the mIB-KEMs [2, 3] were suggested with formal security proofs using random oracle heuristics. These schemes also achieved $O(|\mathsf{S}|)$ ciphertexts and $O(1)$ private keys. Recently, Sakai and Furukawa [18] proposed a new mIB-KEM which uses a exponent inversion paradigm. At first sight, their construction appears to achieve $O(1)$ ciphertexts and $O(1)$ private keys, but the decryption algorithm requires $|\mathsf{S}|$ elements in the public parameters. It means that sender needs to transmit the $|\mathsf{S}|$ elements for decryption together with ciphertext, or each user requires to store all the elements in the public parameters (although the elements could be stored in any public device). This leads their scheme [18] to obtain $O(|\mathsf{S}|)$ ciphertexts or $O(n)$ private keys, where n is the total number of users.

To suggest a secure mIB-KEM without random oracles, the constructions [11, 12, 1] employed the structure of key delegation in HIBE schemes [20, 4, 5]. In [11], Chatterjee et al. presented a mIB-KEM that has $O(|\mathsf{S}|)$ ciphertexts and $O(1)$ private keys, with the security proof in the selective-ID security model. Later, Chatterjee and Sarkar [12] proposed a mIB-KEM (secure in the selective-ID model) that uses a publicly computable surjective function to reduce an identity space to a set $\{1, \ldots, N\}$. The authors demonstrated that their scheme [12] could obtain sublinear-size ciphertexts and private keys at the same time (unfortunately, this scheme has a security flaw although it achieves good performance). Same authors [12] presented another mIB-KEM secure in the full model, but the security reduction has suffered from an exponential security degradation. Recently, Abdalla et at. [1] examined a different variant of delegation structures in HIBE schemes [17, 5] and used the variant to introduce so-called "wicked IBE" which yields an mIB-KEM. However, the resulting instantiations of mIB-KEM do not provide sublinear-size ciphertexts and private keys, simultaneously.

2. PRELIMINARIES

2.1 Multi-receiver Identity-Based Key Encapsulation Mechanism

We describe the definition of multi-receiver Identity-Based Key Encapsulation Mechanism (mIB-KEM) [12] as below.

Setup$(1^k, n)$ takes as input a security parameter 1^k and the number of total users n. and outputs the public parameters PP and the master key MK.

KeyGen$(\mathsf{ID}, \mathsf{MK}, \mathsf{PP})$ takes an identity $\mathsf{ID} \in \mathcal{ID}$, the master key MK, and the public parameters PP as input. It outputs a private key d_{ID} for ID.

Encapsulate$(\mathsf{S}, \mathsf{PP})$ takes a set S of identities and the public parameters PP as input, and outputs a pair (Hdr, K) where Hdr is the header and $K \in \mathcal{K}$ is a message encryption key, often called the broadcast ciphertext.

Let M be a message to be broadcast to the set S and let C_M be the encryption of M under the symmetric key K. A broadcast massage is $(\mathsf{S}, \mathsf{Hdr}, C_M)$, where the pair $(\mathsf{S}, \mathsf{Hdr})$ is often called the full header and C_M is often called the broadcast body.

Decapsulate$(d_{\mathsf{ID}}, \mathsf{S}, \mathsf{Hdr}, \mathsf{PP})$ takes as input the private key d_{ID} for ID, a receiver set S, a header Hdr, and the public parameters PP. If $\mathsf{ID} \in \mathsf{S}$, the algorithm outputs the message encryption key $K \in \mathcal{K}$, which is used to decrypt C_M and obtain the message M.

For correctness, we require that for a receiver set S and $\mathsf{ID} \in \mathsf{S}$, if $(\mathsf{PP}, \mathsf{MK}) \overset{R}{\leftarrow} \mathtt{Setup}(1^k, n)$, $d_{\mathsf{ID}} \leftarrow \mathtt{KeyGen}(\mathsf{ID}, \mathsf{MK}, \mathsf{PP})$, and $(\mathsf{Hdr}, K) \overset{R}{\leftarrow} \mathtt{Encapsulate}(\mathsf{S}, \mathsf{PP})$, then we have that $\mathtt{Decapsulate}(d_{\mathsf{ID}}, \mathsf{S}, \mathsf{Hdr}, \mathsf{PP}) = K$.

Next, to describe the chosen ciphertext security for mIB-KEM, we define the following game between an attacker $\mathcal{A}$ and a challenger $\mathcal{C}$ as in [12]. Both $\mathcal{A}$ and $\mathcal{C}$ are provided with n, the total number of users, as input.

Init: $\mathcal{A}$ outputs a set S^* of identities that it intends to attack.

Setup: $\mathcal{C}$ runs $\mathtt{Setup}(1^k, n)$ to obtain the public parameters PP and the master key MK. It gives $\mathcal{A}$ the public parameters PP.

Phase 1: $\mathcal{A}$ adaptively issues queries $q_1, \ldots, q_m$ where each is one of

1. Private key query on ID where $\mathsf{ID} \notin \mathsf{S}^*$. $\mathcal{C}$ runs algorithm $\mathtt{KeyGen}(\mathsf{ID}, \mathsf{MK}, \mathsf{PP})$ to obtain a private key d_{ID}. It returns d_{ID} to $\mathcal{A}$.

2. Decryption query on $(\mathsf{ID}, \mathsf{S}, \mathsf{Hdr})$ where $\mathsf{S} \subseteq \mathsf{S}^*$ and $\mathsf{ID} \in \mathsf{S}$. $\mathcal{C}$ responds with $\mathtt{Decapsulate}(d_{\mathsf{ID}}, \mathsf{S}, \mathsf{Hdr}, \mathsf{PP})$.

Challenge: $\mathcal{C}$ runs algorithm $\mathtt{Encapsulate}(\mathsf{S}^*, \mathsf{PP})$ to obtain (Hdr^*, K) where $K \in \mathcal{K}$. Next, the challenger picks a random $b \in \{0, 1\}$. If $b = 1$, it sets $K^* = K$. Otherwise, it sets K^* to a random string of length equal to $|K|$. $\mathcal{C}$ gives a challenge ciphertext (Hdr^*, K^*) to $\mathcal{A}$.

Phase 2: $\mathcal{A}$ adaptively issues private key and decryption queries $q_{m+1}, \ldots, q_q$ where each one is:

1. Private key query on ID where $\mathsf{ID} \notin \mathsf{S}^*$. $\mathcal{C}$ responds as in phase 1.

2. Decryption query on $(\mathsf{ID}, \mathsf{S}, \mathsf{Hdr})$ where $\mathsf{S} \subseteq \mathsf{S}^*$ and $\mathsf{ID} \in \mathsf{S}$. The other restriction is that $\mathsf{Hdr} \neq \mathsf{Hdr}^*$. $\mathcal{C}$ responds as in phase 1.

Guess: $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$. $\mathcal{A}$ wins if $b' = b$.

This game above models an attack where all users not in the set S^* collude to try and expose a broadcast message intended only for users in S^*. The attacker in this model is static as in [7]. That is, it chooses S^* and obtains the keys for identities outside of S^*, before it sees the public parameters PP. For short, we call this 'sID security model'. The advantage of $\mathcal{A}$ in breaking a mIB-KEM is defined as

$$\mathsf{Adv}^{\mathsf{mIB\text{-}KEM}}_{\mathcal{A},n} = |\Pr[b = b'] - 1/2|$$

where n is given to both the $\mathcal{C}$ and $\mathcal{A}$ as input.

DEFINITION 1. *A mIB-KEM is said to be $(t, \epsilon, n, q_{\mathsf{ID}}, q_D)$-CCA-secure in the sID security model if for all t-time attackers $\mathcal{A}$ who make q_{ID} private key queries and q_D decryption queries, we have that $\mathsf{Adv}^{\mathsf{mIB\text{-}KEM}}_{\mathcal{A},n} < \epsilon$.*

The game above can be used to define chosen plaintext security for a mIB-KEM if the attacker is not permitted to issue decryption queries. We say that a mIB-KEM is $(t, \epsilon, n, q_{\mathsf{ID}})$-CPA-secure in the sID security model if it is $(t, \epsilon, n, q_{\mathsf{ID}}, 0)$-CCA-secure.

2.2 Bilinear Pairing and Complexity Assumption

We briefly summarize the bilinear pairings and define the $(b+1)$-Bilinear Diffie-Hellman Exponent (BDHE) assumption.

Bilinear Pairing: We follow the notation in [6, 4]. Let $\mathbb{G}$ and $\mathbb{G}_T$ be two (multiplicative) cyclic groups of prime order p. We assume that g is a generator of $\mathbb{G}$. Let $e : \mathbb{G}\times\mathbb{G} \to \mathbb{G}_T$ be a function that has the following properties: 1) Bilinear: for all $u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}$, we have $e(u^a, v^b) = e(u, v)^{ab}$. 2) Non-degenerate: $e(g, g) \neq 1$. 3) Computable: there is an efficient algorithm to compute the map e.

Then, we say that $\mathbb{G}$ is a bilinear group and the map e is a bilinear pairing in $\mathbb{G}$. Note that $e(,)$ is symmetric since $e(g^a, g^b) = e(g, g)^{ab} = e(g^b, g^a)$.

Bilinear Diffie-Hellman Exponent Assumption: The $(b+1)$-BDHE problem in $\mathbb{G}$ is defined as follows: given a $(2b+2)$-tuple $(z, g, g^{\alpha}, \ldots, g^{\alpha^b}, g^{\alpha^{b+2}}, \ldots, g^{\alpha^{2b}}) \in \mathbb{G}^{2b+2}$ as input, compute $e(z, g)^{\alpha^{b+1}} \in \mathbb{G}_T$. An algorithm $\mathcal{A}$ has advantage ϵ in solving $(b+1)$-BDHE in $\mathbb{G}$ if

$$\Pr[\mathcal{A}(z, g, g^{\alpha}, \ldots, g^{\alpha^b}, g^{\alpha^{b+2}}, \ldots, g^{\alpha^{2b}}) = e(z, g)^{\alpha^{b+1}}] \geq \epsilon$$

where the probability is over the random choice of α in $\mathbb{Z}_p$, the random choice of $z \in \mathbb{G}$, and the random bits of $\mathcal{A}$.

Let $g_i = g^{(\alpha^i)}$ and let $\overrightarrow{g}_{\alpha,b} = (g_1, \ldots, g_b, g_{b+2}, \ldots, g_{2b})$. Similarly, we say that an algorithm $\mathcal{B}$ that outputs $b \in \{0, 1\}$ has advantage ϵ in solving the *decision* $(b+1)$-BDHE problem in $\mathbb{G}$ if

$$|\Pr[\mathcal{B}(z, g, \overrightarrow{g}_{\alpha,b},\ e(z, g_{b+1})) = 0] - \Pr[\mathcal{B}(z, g, \overrightarrow{g}_{\alpha,b},\ T) = 0]| \geq \epsilon$$

where the probability is over the random choice of α in $\mathbb{Z}_p$, the random choice of $z \in \mathbb{G}$, the random choice of $T \in \mathbb{G}_T$, and the random bits of $\mathcal{B}$.

DEFINITION 2. *We say that the (decision) $(t, \epsilon, b+1)$-BDHE assumption holds in $\mathbb{G}$ if no t-time algorithm has advantage at least ϵ in solving the (decision) $(b+1)$-BDHE problem in $\mathbb{G}$.*

3. CRYPTANALYSIS OF THE CHATTERJEE AND SARKAR'S mIB-KEM

We review the mIB-KEM suggested by Chatterjee and Sarkar [12], and analyze the security flaws in their scheme.

3.1 Review of the Chatterjee and Sarkar's mIB-KEM

Let $\mathbb{G}$ and $\mathbb{G}_T$ be groups of prime order p, and let $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be the bilinear map.

Setup$(1^k, n)$: The algorithm picks a random generator $g \in \mathbb{G}$. It selects a random $\alpha \in \mathbb{Z}_p^*$ and sets $g_1 = g^{\alpha}$. It picks random elements $g_2, g_3 \in \mathbb{G}$ and a random vector $\vec{U} = (u_1, \ldots, u_a)$ with entries in $\mathbb{G}$. The public parameters PP (with the description of ($\mathbb{G}$, $\mathbb{G}_T$, e, p)) and the master key MK are given by

$$\mathsf{PP} = (g, g_1, g_2, g_3, \vec{U}, H), \qquad \mathsf{MK} = g_2^{\alpha}$$

where H is a publicly computable surjective function such that $H : \mathbb{Z}_p^* \to \{1, \ldots, a\}$.

KeyGen$(\mathsf{ID}, \mathsf{MK}, \mathsf{PP})$: Given an identity $\mathsf{ID} \in \mathbb{Z}_p^*$, the algorithm computes $H(\mathsf{ID}) = v$ where $1 \leq v \leq a$. It picks a random $r \in \mathbb{Z}_p$ and output

$$d_{\mathsf{ID}} = (\ g_2^{\alpha} \cdot (g_3 \cdot u_v^{\mathsf{ID}})^r,\ g^r,\ u_1^r, \ldots,\ u_{v-1}^r,\ u_{v+1}^r, \ldots,\ u_a^r\ ) \in \mathbb{G}^{a+1}.$$

Encapsulate$(\mathsf{S}, \mathsf{PP})$: A sender chooses a random $s \in \mathbb{Z}_p$ and sets $K = e(g_1, g_2)^s \in \mathbb{G}_T$. Next, the sender partitions the set S into subsets in the following manner.

Let $H(\mathsf{S}) = \{j_1, \ldots, j_k\}$ be the set of distinct indices obtained by applying the function H to the elements in S. For $i = 1, \ldots, k$, let $\{s_{i,1}, \ldots, s_{i,\tau_i}\}$ be the subset of all elements in S which map to j_i. Let $\tau = \max_{1 \leq i \leq k}(\tau_i)$. We view S as a $k \times \tau$ matrix having entries $s_{i,j}$ where $1 \leq i \leq k$ and $1 \leq j \leq \tau_i$. For $1 \leq j \leq \tau$, define the set S_j to be the j-th column of this matrix. Then S is a disjoint union of $\mathsf{S}_1, \ldots, \mathsf{S}_\tau$ and for all j, we have $|\mathsf{S}_j| = |H(\mathsf{S}_j)|$ (it means H is injective on S_j).
Then, the sender sets the header as

$$\mathsf{Hdr} = (\ (g_3 \cdot \Pi_{\mathsf{ID}\in\mathsf{S}_1}(u_{H(\mathsf{ID})})^{\mathsf{ID}})^s, \ldots,\ (g_3 \cdot \Pi_{\mathsf{ID}\in\mathsf{S}_\tau}(u_{H(\mathsf{ID})})^{\mathsf{ID}})^s,\ g^s\ ) \in \mathbb{G}^{\tau+1}.$$

The sender broadcasts $(\mathsf{S}_1, \ldots, \mathsf{S}_\tau, \mathsf{Hdr}, C_M)$, where C_M is an encrypted message under the K using a symmetric key cipher.

Decapsulate$(d_{\mathsf{ID}}, \mathsf{S}, \mathsf{Hdr})$: Let $\mathsf{Hdr} = (A_1, \ldots, A_\tau, B)$. Assume a receiver with identity ID belongs to the subset S_j and $H(\mathsf{ID}) = v$. To decrypt the Hdr, it is sufficient for him to obtain (S_j, A_j, B) where $\mathsf{ID} \in \mathsf{S}_j$. The receiver decrypts the Hdr using his private key $d_{\mathsf{ID}} = (d_{\mathsf{ID},1}, d_{\mathsf{ID},2}, k_{\mathsf{ID},1}, \ldots, k_{\mathsf{ID},v-1}, k_{\mathsf{ID},v+1}, \ldots, k_{\mathsf{ID},a})$ as follows:

$$K = e(d_{\mathsf{ID},1} \cdot \Pi_{\substack{\mathsf{ID}'\in\mathsf{S}_j \\ \mathsf{ID}'\neq\mathsf{ID}}} (k_{\mathsf{ID},H(\mathsf{ID}')})^{\mathsf{ID}'},\ B)\ /\ e(A_j,\ d_{\mathsf{ID},2}).$$

Correctness of the decapsulation above, which we omit here, can be showed by the similar calculation to that of Section 4.1.

3.2 Security Analysis

We describe how a revoked user can decrypt the Hdr successfully. Let ID' be the identity of a revoked user. Let $\mathsf{S}_1, \mathsf{S}_2$ be the subsets of receivers such that $\mathsf{ID}_1 \in \mathsf{S}_1$ and $\mathsf{ID}_2 \in \mathsf{S}_2$. From the partition of the receiver set, we can assume the case where $H(\mathsf{ID}_1) = H(\mathsf{ID}_2) = H(\mathsf{ID}') = v$, because the function $H : \mathbb{Z}_p^* \to \{1, \ldots, a\}$ is surjective. Then, the A_1 and A_2 elements in the $\mathsf{Hdr} = (A_1, A_2, \ldots, A_\tau, B)$ are constructed as

$$A_1 = (g_3 \cdots u_v^{\mathsf{ID}_1} \cdots)^s, \qquad A_2 = (g_3 \cdots u_v^{\mathsf{ID}_2} \cdots)^s$$

for some (unknown) $s \in \mathbb{Z}_p$. Let $l = (\mathsf{ID}_1 - \mathsf{ID}')/(\mathsf{ID}' - \mathsf{ID}_2) \in \mathbb{Z}_p^*$, which is computable since $\mathsf{ID}' \neq \mathsf{ID}_1$ and $\mathsf{ID}' \neq \mathsf{ID}_2$. The revoked user then computes $A' = (A_1 A_2^l)^{1/(l+1)}$ where $(l+1)$ would become zero with negligible probability. Observe that

$$\begin{aligned} A' &= (g_3 \cdots u_v^{\mathsf{ID}_1} \cdot g_3^l \cdots u_v^{\mathsf{ID}_2 \cdot l} \cdots)^{s/(l+1)} \\ &= (g_3^{l+1} \cdots u_v^{\mathsf{ID}_1 + \mathsf{ID}_2 \cdot l} \cdots)^{s/(l+1)} \\ &= (g_3 \cdots u_v^{(\mathsf{ID}_1 + \mathsf{ID}_2 \cdot l)/(l+1)} \cdots)^s \\ &= (g_3 \cdots u_v^{\mathsf{ID}'} \cdots)^s. \end{aligned}$$

Since the set information about S_1 and S_2 is transmitted along with the Hdr, the revoked user can easily know the exponential value of each u_k for $k = 1, \ldots, v-1, v+1, \ldots, a$ (if necessary). Recall that the private key for the revoked user ID' is

$$d_{\mathsf{ID}'} = (\ g_2^\alpha \cdot (g_3 \cdot u_v^{\mathsf{ID}'})^r,\ g^r,\ u_1^r, \ldots,\ u_{v-1}^r,\ u_{v+1}^r, \ldots,\ u_a^r\ )$$

which can be used to decrypt the Hdr using the elements (A', B), where B is the element in the Hdr.

We consider the simpler case where $H(\mathsf{ID}_1) = H(\mathsf{ID}') = v$. Then, the A_1 and A_2 elements are computed as

$$A_1 = (g_3 \cdots u_v^{\mathsf{ID}_1} \cdots)^s, \qquad A_2 = (g_3 \cdots u_{v-1}^{\mathsf{ID}_3} \cdot u_{v+1}^{\mathsf{ID}_4} \cdots)^s$$

for some identities $\mathsf{ID}_3, \mathsf{ID}_4 \in \mathsf{S}_2$. (Here, the A_2 could be constructed differently, but the important point is that S_2 does not include an identity $\mathsf{ID} \in \mathbb{Z}_p^*$ such that $H(\mathsf{ID}) = H(\mathsf{ID}') = v$.) In this case, let $l' = (\mathsf{ID}_1 - \mathsf{ID}')/\mathsf{ID}' \in \mathbb{Z}_p$. The revoked user computes $A'' = (A_1 A_2^{l'})^{1/(l'+1)}$. By the similar calculation to that above, we can see that $A'' = (g_3 \cdots u_v^{\mathsf{ID}'} \cdots)^s$ for some unknown $s \in \mathbb{Z}_p$. This allows the revoked user to decrypt the Hdr successfully.

In [12], the authors focused only on the impossibility of revoked users for building a private key suitable for decryption. However, the observation above shows that a revoked user could generate a valid component of the Hdr, and successfully recover the message encryption key, given that there is some possible collision of the function H. As shown in the above analysis, this is because the same element g_3 is used for all the partitioned subsets. Thus, in order to avoid such an attack, one natural solution is to use different elements for each subset. We will present this solution in the next section.

Chatterjee and Sarkar [12] suggested a CCA-secure mIB-KEM which is based on the CPA-secure scheme. Unfortunately, we can show the resulting scheme is also insecure against the attacks described above in the same manner.

4. CPA-SECURE mIB-KEM

In this section we present a new CPA-secure mIB-KEM which overcomes the security leak identified in the previous section. The crux of our method is to use different elements associated with subsets of receivers, which can prevent such attacks from occurring. To relate an identity to one of these additional elements, we need another publicly computable surjective function, denoted by H_1 here.

As opposed to the previous scheme [12], our construction imposes a priori maximum number n of receivers as the input of `Setup` algorithm. For now, we assume identities are elements of $\mathbb{Z}_p^*$, but as noted in [4] we can extend the domain to all of $\{0,1\}^*$ by hashing each identity ID using a collision resistant hash function $H : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$. Note that `Decapsulate` algorithm below does not require any elements of public parameters as input.

4.1 Scheme

Let $\mathbb{G}$ and $\mathbb{G}_T$ be groups of prime order p, and let $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ be the bilinear map.

Setup$(1^k, n)$: The algorithm picks a random generator $g \in \mathbb{G}$. It selects a random $\alpha \in \mathbb{Z}_p^*$ and sets $g_1 = g^\alpha$. It picks random elements $x_0, x_1, \ldots, x_a, y_1, \ldots, y_b \in \mathbb{G}$. The public parameters PP (with the description of $(\mathbb{G}, \mathbb{G}_T, e, p)$) and the master key MK are given by

$$\mathsf{PP} = (g, g_1, x_0, x_1, \ldots, x_a, y_1, \ldots, y_b, H_1, H_2), \qquad \mathsf{MK} = \alpha$$

where H_1 and H_2 are publicly computable surjective functions such that $H_1 : \mathbb{Z}_p^* \rightarrow \{1, \ldots, a\}$ and $H_2 : \mathbb{Z}_p^* \rightarrow \{1, \ldots, b\}$.

KeyGen$(\mathsf{ID}, \mathsf{MK}, \mathsf{PP})$: Given an identity $\mathsf{ID} \in \mathbb{Z}_p^*$, the algorithm finds two values u, v (where $1 \leq u \leq a$ and $1 \leq v \leq b$) such that $H_1(\mathsf{ID}) = u$ and $H_2(\mathsf{ID}) = v$. If there already exists an identity $\mathsf{ID}' \in \mathbb{Z}_p^*$ such that $H_1(\mathsf{ID}') = u$ and $H_2(\mathsf{ID}') = v$, the KGC aborts. Otherwise, it picks a random $r \in \mathbb{Z}_p$ and set the private key for ID as

$$\begin{aligned} d_{\mathsf{ID}} = (\ & x_0^\alpha \cdot (x_u \cdot y_v^{\mathsf{ID}})^r,\ g^r, \\ & y_1^r, \ldots,\ y_{v-1}^r,\ y_{v+1}^r, \ldots,\ y_b^r\ ) \in \mathbb{G}^{b+1}. \end{aligned}$$

Encapsulate$(\mathsf{S},\ \mathsf{PP})$: A sender chooses a random $s \in \mathbb{Z}_p$ and set $K = e(x_0, g_1)^s \in \mathbb{G}_T$. Wlog, we can assume the set S is divided into subsets $\mathsf{S}_1, \ldots, \mathsf{S}_a$ after computing $H_1(\mathsf{ID})$ and $H_2(\mathsf{ID})$ for $\mathsf{ID} \in \mathsf{S}$. Then, for all $\mathsf{ID} \in \mathsf{S}_i$, we have $H_1(\mathsf{ID}) = i$. Set the header as

$$\begin{aligned} \mathsf{Hdr} = (\ & (x_1 \cdot \Pi_{\mathsf{ID} \in \mathsf{S}_1} (y_{H_2(\mathsf{ID})})^{\mathsf{ID}})^s, \ldots, \\ & (x_a \cdot \Pi_{\mathsf{ID} \in \mathsf{S}_a} (y_{H_2(\mathsf{ID})})^{\mathsf{ID}})^s,\ g^s\ ) \in \mathbb{G}^{a+1}. \end{aligned}$$

The algorithm outputs the pair (Hdr, K). Then, the sender broadcasts $(\mathsf{S}, \mathsf{Hdr}, C_M)$, where C_M is an encrypted message under the K using a symmetric key cipher.

Decapsulate$(d_{\mathsf{ID}},\ \mathsf{S},\ \mathsf{Hdr})$: Assume a receiver with identity ID is assigned to index u, v such that $H_1(\mathsf{ID}) = u$ and $H_2(\mathsf{ID}) = v$. The receiver decrypts the Hdr using his private key $d_{\mathsf{ID}} = (d_{\mathsf{ID},1}, d_{\mathsf{ID},2}, k_{\mathsf{ID},1}, \ldots, k_{\mathsf{ID},v-1}, k_{\mathsf{ID},v+1}, \ldots, k_{\mathsf{ID},b})$. Let $\mathsf{Hdr} = (A_1, \ldots, A_a, B)$. Then, output

$$K = e(d_{\mathsf{ID},1} \cdot \Pi_{\substack{\mathsf{ID}' \in \mathsf{S}_u \\ \mathsf{ID}' \neq \mathsf{ID}}} (k_{\mathsf{ID}, H_2(\mathsf{ID}')})^{\mathsf{ID}'},\ B)\ /\ e(A_u,\ d_{\mathsf{ID},2}).$$

Correctness: Assuming the Hdr is well-formed, the correctness of the decapsulation is checked as follows:

$$\begin{aligned} K &= e(d_{\mathsf{ID},1} \cdot \Pi_{\substack{\mathsf{ID}' \in \mathsf{S}_u \\ \mathsf{ID}' \neq \mathsf{ID}}} (k_{\mathsf{ID}, H_2(\mathsf{ID}')})^{\mathsf{ID}'},\ B)\ /\ e(A_u,\ d_{\mathsf{ID},2}) \\ &= e(x_0^\alpha \cdot (x_u \cdot y_v^{\mathsf{ID}})^r \cdot \Pi_{\substack{\mathsf{ID}' \in \mathsf{S}_u \\ \mathsf{ID}' \neq \mathsf{ID}}} (y_{H_2(\mathsf{ID}')}^r)^{\mathsf{ID}'},\ g^s)\ / \\ &\qquad\qquad e((x_u \cdot \Pi_{\mathsf{ID}' \in \mathsf{S}_u} (y_{H_2(\mathsf{ID}')})^{\mathsf{ID}'})^s,\ g^r) \\ &= e(x_0^\alpha \cdot (x_u \cdot \Pi_{\mathsf{ID}' \in \mathsf{S}_u} (y_{H_2(\mathsf{ID}')})^{\mathsf{ID}'})^r,\ g^s)\ / \\ &\qquad\qquad e((x_u \cdot \Pi_{\mathsf{ID}' \in \mathsf{S}_u} (y_{H_2(\mathsf{ID}')})^{\mathsf{ID}'})^s,\ g^r) \\ &= e(x_0, g_1)^s. \end{aligned}$$

REMARK: As stated in [12], the outputs of two surjective functions H_1 and H_2 are expected to be uniformly distributed, so that the entire elements of x_i and y_j could be used. However, H_1 and H_2 do not act as random oracles in our security proofs.

4.2 Security

The CPA-security of the mIB-KEM above is proven under the decision $(b+1)$-BDHE assumption.

THEOREM 1. *Suppose that the decision $(t, \epsilon, b+1)$-BDHE assumption holds in $\mathbb{G}$. Then the previous mIB-KEM is $(t', \epsilon, n, q_{ID})$-CPA-secure in the sID security model for any positive integers n, b and $t' < t - \Theta(\tau b n)$, where τ is the maximum time for an exponentiation in $\mathbb{G}$.*

PROOF. Suppose there exists an adversary $\mathcal{A}$ which has advantage ϵ in attacking the mIB-KEM. We want to build an algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to solve the decision $(b+1)$-BDHE problem in $\mathbb{G}$. For a generator $g \in \mathbb{G}$ and $\alpha \in \mathbb{Z}_p$, let $g_i = g^{(\alpha^i)} \in \mathbb{G}$. On input $(z, g, g_1, \ldots, g_b, g_{b+2}, \ldots, g_{2b}, T)$, $\mathcal{B}$ outputs 1 if $T = e(z, g_{b+1})$ and 0 otherwise. $\mathcal{B}$ works by interacting with $\mathcal{A}$ as follows:

Init: $\mathcal{A}$ outputs a set S^* that it intends to attack.

Setup: After deciding two publicly computable surjective functions $H_1 : \mathbb{Z}_p^* \to \{1, \ldots, a\}$ and $H_2 : \mathbb{Z}_p^* \to \{1, \ldots, b\}$, $\mathcal{B}$ divides the challenge set S^* into subsets $\mathsf{S}_1^*, \ldots, \mathsf{S}_a^*$. It depends on the values $H_1(\mathsf{ID})$ and $H_2(\mathsf{ID})$ for $\mathsf{ID} \in \mathsf{S}^*$. Next, $\mathcal{B}$ selects a random $\rho \in \mathbb{Z}_p$ and set $x_0 = g_b \cdot g^{\rho}$. It also picks random $\gamma_1, \ldots, \gamma_b, \delta_1, \ldots, \delta_a \in \mathbb{Z}_p$. It sets $y_i = g^{\gamma_i} g_i$ for $i = 1, \ldots, b$ and sets $x_j = g^{\delta_j} \cdot (\Pi_{\mathsf{ID} \in \mathsf{S}_j^*} (g_{H_2(\mathsf{ID})})^{\mathsf{ID}})^{-1}$ for $j = 1, \ldots, a$. Finally, $\mathcal{B}$ gives $\mathcal{A}$ the public parameters

$$\mathsf{PP} = (\ g, g_1, x_0, x_1, \ldots, x_a, y_1, \ldots, y_b, H_1, H_2\ ).$$

Since $\rho, \{\gamma_i\}$, and $\{\delta_j\}$ values are chosen uniformly at random, this public key has an identical distribution to that in the actual construction.

Phase 1: $\mathcal{B}$ needs to generate private keys d_{ID} for $\mathsf{ID} \notin \mathsf{S}^*$. Consider a private key for ID such that $H_1(\mathsf{ID}) = u$ and $H_2(\mathsf{ID}) = v$. Wlog, we can assume the pair (u, v) does not belong to the set $\{(H_1(\mathsf{ID}), H_2(\mathsf{ID})) : \mathsf{ID} \in \mathsf{S}^*\}$. $\mathcal{B}$ picks a random $r \in \mathbb{Z}_p$. Let $r' = r - \alpha^{(b+1-v)} /\ \mathsf{ID}$. (Recall that $\mathsf{ID} \in \mathbb{Z}_p^*$) $\mathcal{B}$ generates the private key d_{ID} for ID as

$$(\ x_0^{\alpha} \cdot (x_u \cdot y_v^{\mathsf{ID}})^{r'},\ g^{r'},\ y_1^{r'}, \ldots,\ y_{v-1}^{r'},\ y_{v+1}^{r'}, \ldots,\ y_b^{r'}\ )$$

which is a properly distributed private key for ID due to the randomness r. We show that $\mathcal{B}$ can compute all components of this private key given the values that it knows. To generate the first component of the private key, observe that

$$\begin{aligned}
&(x_u \cdot y_v^{\mathsf{ID}})^{r'} \\
&= (g^{\delta_u} (\Pi_{\mathsf{ID}' \in \mathsf{S}_u^*} (g_{H_2(\mathsf{ID}')})^{\mathsf{ID}'})^{-1} \cdot (g^{\gamma_v} g_v)^{\mathsf{ID}})^{r'} \\
&= (g^{\delta_u} (\Pi_{\mathsf{ID}' \in \mathsf{S}_u^*} (g_{H_2(\mathsf{ID}')})^{\mathsf{ID}'})^{-1} \cdot (g^{\gamma_v} g_v)^{\mathsf{ID}})^{r} \cdot \\
&\qquad (g^{\delta_u} (\Pi_{\mathsf{ID}' \in \mathsf{S}_u^*} (g_{H_2(\mathsf{ID}')})^{\mathsf{ID}'})^{-1} \cdot (g^{\gamma_v} g_v)^{\mathsf{ID}})^{-\alpha^{b+1-v}/\mathsf{ID}} \\
&= \ldots (g_{b+1-v}^{(\delta_u + \gamma_v \mathsf{ID})} (\Pi_{\mathsf{ID}' \in \mathsf{S}_u^*} (g_{b+1-v+H_2(\mathsf{ID}')})^{\mathsf{ID}'})^{-1})^{-1/\mathsf{ID}} \cdot g_{b+1}^{-1}.
\end{aligned}$$

Note that $H_2(\mathsf{ID}) = v$. Since $\mathsf{ID} \notin \mathsf{S}^*$, we see that $\mathsf{ID} \notin \mathsf{S}_u^*$ and thus $H_2(\mathsf{ID}') - v \neq 0$ for any $\mathsf{ID}' \in \mathsf{S}_u^*$. Since $x_0^{\alpha} = g_{b+1} \cdot g_1^{\rho}$, the first component in the private key can be computed as

$$\begin{aligned}
&g_1^{\rho} \cdot (g^{\delta_u} (\Pi_{\mathsf{ID}' \in \mathsf{S}_u^*} (g_{H_2(\mathsf{ID}')})^{\mathsf{ID}'})^{-1} \cdot (g^{\gamma_v} g_v)^{\mathsf{ID}})^{r} \cdot \\
&\qquad (g_{b+1-v}^{(\delta_u + \gamma_v \mathsf{ID})} \cdot (\Pi_{\mathsf{ID}' \in \mathsf{S}_u^*} (g_{b+1-v+H_2(\mathsf{ID}')})^{\mathsf{ID}'})^{-1})^{-1/\mathsf{ID}}
\end{aligned}$$

where the unknown term g_{b+1} is canceled out. The other terms $g^{r'}$ and $y_i^{r'}$ are computable since $g^{r'} = g^{r}(g_{b+1-v})^{-1/\mathsf{ID}}$ and $y_i^{r'} = (g^{\gamma_i} g_i)^{r} (g_{b+1-v}^{\gamma_i} \cdot g_{b+1-v+i})^{-1/\mathsf{ID}}$ for $i = 1, \ldots, v-1, v+1, \ldots, b$. Since $i \neq v$, these values do not require knowledge of g_{b+1}.

Challenge: To generate a challenge (Hdr^*, K^*) under the receiver set S^*, $\mathcal{B}$ sets

$$\mathsf{Hdr}^* = (\ z^{\delta_1 + \Sigma_{\mathsf{ID} \in \mathsf{S}_1^*} \mathsf{ID} \cdot \gamma_{H_2(\mathsf{ID})}},\ \ldots,\ z^{\delta_a + \Sigma_{\mathsf{ID} \in \mathsf{S}_a^*} \mathsf{ID} \cdot \gamma_{H_2(\mathsf{ID})}},\ z\ )$$

and $K^* = T \cdot e(g_1, z^{\rho})$, where z and T are input values given to $\mathcal{B}$. Observe that if $z = g^c$ for some (unknown) $c \in \mathbb{Z}_p$, then

$$\begin{aligned}
&z^{\delta_i + \Sigma_{\mathsf{ID} \in \mathsf{S}_i^*} \mathsf{ID} \cdot \gamma_{H_2(\mathsf{ID})}} \\
&= (g^{\delta_i} (\Pi_{\mathsf{ID} \in \mathsf{S}_i^*} (g_{H_2(\mathsf{ID})})^{\mathsf{ID}})^{-1} \cdot \Pi_{\mathsf{ID} \in \mathsf{S}_i^*} (g^{\gamma_{H_2(\mathsf{ID})}} g_{H_2(\mathsf{ID})})^{\mathsf{ID}})^{c} \\
&= (x_i \cdot \Pi_{\mathsf{ID} \in \mathsf{S}_i^*} (y_{H_2(\mathsf{ID})})^{\mathsf{ID}})^{c}
\end{aligned}$$

for $i = 1, \ldots, a$. If $T = e(z, g_{b+1})$ then $K^* = e(x_0, g_1)^c$ and thus (Hdr^*, K^*) is a valid challenge to $\mathcal{A}$ for the receiver set S^*. On the other hand, when T is uniform and independent in $\mathbb{G}_T$, then Hdr^* is independent of K^* in the adversary's view.

Phase 2: $\mathcal{A}$ issues private key queries. $\mathcal{B}$ responds as before.

Guess: $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$. If $b' = 1$ then it indicates $T = e(z, g_{b+1})$. Otherwise, it indicates $T \neq e(z, g_{b+1})$.

When T is random in $\mathbb{G}_T$ then $\Pr[\mathcal{B}(z, g, \overrightarrow{g}_{\alpha,b}, T) = 0] = 1/2$. When $T = e(z, g_{b+1})$, $\mathcal{B}$ replied with a valid challenge (Hdr^*, K^*). Then $|\Pr[b = b'] - 1/2| \geq \epsilon$. Therefore, $\mathcal{B}$ has that

$$\begin{aligned}
&|\Pr[\mathcal{B}(z, g, \overrightarrow{g}_{\alpha,b}, e(z, g_{b+1})) = 0] \\
&\qquad\qquad - \Pr[\mathcal{B}(z, g, \overrightarrow{g}_{\alpha,b}, T) = 0]| \geq \epsilon.
\end{aligned}$$

This completes the proof of Theorem 1. □

4.3 Performance

Let n be the total number of users the mIB-KEM can handle, and let S be the set of receivers. Recall that the values a, b are derived from the two publicly computable surjective functions $H_1 : \mathbb{Z}_p^* \to \{1, \ldots, a\}$ and $H_2 : \mathbb{Z}_p^* \to \{1, \ldots, b\}$.

Table 1 shows the performance comparison between the previous mIB-KEMs [2, 11, 1] and ours for $n = ab$, in terms of header(Hdr) size, private key size, decryption cost, and public parameters (PP) size. Note that the mID-KEM in [1] is obtained from so-called wicked identity-based encryption based on the Boneh, Boyen, and Goh's HIBE scheme [5]. Unlike the previous schemes [2, 11, 1], our mIB-KEM provides a tradeoff between the Hdr and private key and PP sizes. With appropriate parametrization, our scheme could have sub-linear size Hdr and private keys simultaneously. This sub-linearity depends on the choice of a and the number $|\mathsf{S}|$ of receivers. For example, let $a = b = \sqrt{n}$. When $|\mathsf{S}| > a$ our scheme has sub-linear size Hdr and private keys, but when $|\mathsf{S}| \leq a$ the Hdr size could become $(|\mathsf{S}|+1)$ elements in $\mathbb{G}$ only in the worst case.

We notice that the above mIB-KEMs [2, 11, 1] including ours are all proven secure in the sID security model. Until

Table 1: Performance Comparison of mIB-KEMs for $n(=ab)$

	Hardness Assumption	Random Oracle	Hdr size	Private Key size	Decryption cost	PP size
BSS [2]	DBDH	Yes	$(\lvert\mathsf{S}\rvert+1)\ \mathtt{G}$	$1\ \mathtt{G}$	$2\ \mathtt{p}$	$3\ \mathtt{G}$
CS [11]	DBDH	No	$(\lvert\mathsf{S}\rvert+1)\ \mathtt{G}$	$2\ \mathtt{G}$	$2\ \mathtt{p} + b\ \mathtt{G_e}$	$(b+4)\ \mathtt{G}$
AKN [1]	DBDHE	No	$2\ \mathtt{G}$	$n(n+1)\ \mathtt{G}$	$2\ \mathtt{p} + \lvert\mathsf{S}\rvert\ \mathtt{G_e}$	$(n+3)\ \mathtt{G}$
Ours	DBDHE	No	$(a+1)\ \mathtt{G}$	$(b+1)\ \mathtt{G}$	$2\ \mathtt{p} + b\ \mathtt{G_e}$	$(a+b+3)\ \mathtt{G}$

$\mathtt{G}$: element in $\mathbb{G}$, $\mathtt{p}$: pairing in $\mathbb{G}$, $\mathtt{G_e}$: exponentiation in $\mathbb{G}$.

now, two mIB-KEMs [3, 12] are suggested to obtain security in the full model, but these schemes have suffered from an exponential security degradation in N, where N is the number of "target" users.

5. CCA-SECURE mIB-KEM

In this section we propose a CCA-secure mIB-KEM by applying the ideas of hash-based method (so called "BMW transformation") in [9] to our CPA-secure construction. Unlike the signature-based method [10] and message authentication code (MAC)-based method [8], the BMW transformation does not need to attach a one-time signature or a MAC to a ciphertext. In particular, the BMW transformation is more suitable for key encapsulation than other methods [10, 8]. To employ the BMW transformation, we need a family of collision resistant hash functions $H_k : \mathbb{G} \to \mathbb{Z}_p$ indexed by $k \in \mathcal{K}$. We say that a family of hash functions is (t, ϵ)-collision resistant if no t-time adversary is able to find two distinct values x, y such that $H_k(x) = H_k(y)$ with probability at least ϵ.

5.1 Scheme

Let $\mathbb{G}$ and $\mathbb{G}_T$ be groups of prime order p, and let $e : \mathbb{G}\times\mathbb{G} \to \mathbb{G}_T$ be the bilinear map. We note that `Decapsulate` algorithm requires the public parameters PP as input, as opposed to the CPA-secure scheme in the previous section.

Setup($1^k, n$): In addition to the setup procedure for the CPA-secure scheme, the algorithm picks a random element $h \in \mathbb{G}$ and selects a random hash key $k \in \mathcal{K}$ for hash function H. The public parameters PP (with the description of ($\mathbb{G}$, $\mathbb{G}_T$, e, p, H_k)) and the master key MK are given by

$$\mathsf{PP} = (g, g_1, h, x_0, x_1, \ldots, x_a, y_1, \ldots, y_b, H_1, H_2), \quad \mathsf{MK} = \alpha.$$

KeyGen($\mathsf{ID}, \mathsf{MK}, \mathsf{PP}$): The private key for $\mathsf{ID} \in \mathbb{Z}_p$ is computed as follows: as before, find two values u, v (where $1 \le u \le a$ and $1 \le v \le b$) such that $H_1(\mathsf{ID}) = u$ and $H_2(\mathsf{ID}) = v$. Pick a random $r \in \mathbb{Z}_p$ and set the private key for ID as

$$\begin{aligned} d_i = (\ & x_0^{\alpha} \cdot (x_u \cdot y_v^{\mathsf{ID}})^r,\ h^r,\ g^r, \\ & y_1^r, \ldots,\ y_{v-1}^r,\ y_{v+1}^r, \ldots,\ y_b^r\ ) \in \mathbb{G}^{b+2}. \end{aligned}$$

Encapsulate(S, PP): A sender chooses a random $s \in \mathbb{Z}_p$ and set $K = e(x_0, g_1)^s \in \mathbb{G}_T$. Next, the sender computes g^s and $\mu = H_k(g^s)$. A header (Hdr) is generated as

$$\begin{aligned} \mathsf{Hdr} = (\ & (x_1 \cdot h^{\mu} \cdot \Pi_{\mathsf{ID}\in\mathsf{S}_1}(y_{H_2(\mathsf{ID})})^{\mathsf{ID}})^s, \ldots, \\ & (x_a \cdot h^{\mu} \cdot \Pi_{\mathsf{ID}\in\mathsf{S}_a}(y_{H_2(\mathsf{ID})})^{\mathsf{ID}})^s,\ g^s\ ) \in \mathbb{G}^{a+1}. \end{aligned}$$

The algorithm outputs the pair (Hdr, K). The sender broadcasts ($\mathsf{S}, \mathsf{Hdr}, C_M$).

Decapsulate(d_{ID}, S, Hdr, PP): As before, a receiver with identity ID is assigned to index u, v such that $H_1(\mathsf{ID}) = u$ and $H_2(\mathsf{ID}) = v$. The receiver decrypts the Hdr using his private key $d_{\mathsf{ID}} = (d_{\mathsf{ID},1}, d_{\mathsf{ID},2}, d_{\mathsf{ID},3},\ k_{\mathsf{ID},1}, \ldots,\ k_{\mathsf{ID},v-1}, k_{\mathsf{ID},v+1}, \ldots, k_{\mathsf{ID},b})$. Let $\mathsf{Hdr} = (A_1, \ldots, A_a, B)$. Compute $\mu' = H_k(B)$ and check that the following equality

$$e(A_u,\ g) = e(x_u \cdot h^{\mu'} \cdot \Pi_{\mathsf{ID}\in\mathsf{S}_u}(y_{H_2(\mathsf{ID})})^{\mathsf{ID}},\ B)$$

holds. If not, output $\perp$. Otherwise, output

$$K = e(d_{\mathsf{ID},1} \cdot d_{\mathsf{ID},2}^{\mu'} \cdot \Pi_{\substack{\mathsf{ID}'\in\mathsf{S}_u \\ \mathsf{ID}'\neq\mathsf{ID}}} (k_{\mathsf{ID},H_2(\mathsf{ID}')})^{\mathsf{ID}'},\ B)\ /\ e(A_u,\ d_{\mathsf{ID},3}).$$

Note that the pair $(d_{\mathsf{ID},1} \cdot d_{\mathsf{ID},2}^{\mu'} \cdot \Pi_{\substack{\mathsf{ID}'\in\mathsf{S}_u \\ \mathsf{ID}'\neq\mathsf{ID}}} (k_{\mathsf{ID},H_2(\mathsf{ID}')})^{\mathsf{ID}'},\ d_{\mathsf{ID},3})$ is chosen from the following distribution

$$(\ x_0^{\alpha} \cdot (x_u \cdot h^{\mu'} \cdot \Pi_{\mathsf{ID}\in\mathsf{S}_u}(y_{H_2(\mathsf{ID})})^{\mathsf{ID}})^{r'}, \quad g^{r'}\ )$$

where r' is uniform in $\mathbb{Z}_p$. We can show that the correctness of decapsulation is checked by the similar calculation to the one in Section 4.1.

To save the pairing computations in the decapsulation, we can use the same technique described in [15]. In that case, the decapsulation algorithm picks a random $w \in \mathbb{Z}_p$ and computes

$$\begin{aligned} d'_{\mathsf{ID},1} &= (d_{\mathsf{ID},1} \cdot d_{\mathsf{ID},2}^{\mu'} \cdot \Pi_{\substack{\mathsf{ID}'\in\mathsf{S}_u \\ \mathsf{ID}'\neq\mathsf{ID}}} (k_{\mathsf{ID},H_2(\mathsf{ID}')})^{\mathsf{ID}'}) \cdot \\ &\qquad\qquad (x_u \cdot h^{\mu'} \cdot \Pi_{\mathsf{ID}\in\mathsf{S}_u}(y_{H_2(\mathsf{ID})})^{\mathsf{ID}})^w, \\ d'_{\mathsf{ID},3} &= d_{\mathsf{ID},3} \cdot g^w. \end{aligned}$$

The algorithm then outputs $K = e(d'_{\mathsf{ID},1}, B)/e(A_u, d'_{\mathsf{ID},3})$. Consequently, this is the same approach to that of [7].

5.2 Security

As opposed to the $(b+1)$-BDHE assumption for the CPA security in Section 4, the CCA security of the above mIB-KEM is based on the $(b+2)$-BDHE assumption.

THEOREM 2. *Suppose that the decision $(t_1, \epsilon_1, b+2)$-BDHE assumption holds in $\mathbb{G}$ and the family of hash function $\{H_k\}$ is (t_2, ϵ_2)-collision resistant. Then the previous mIB-KEM is $(t_3, \epsilon_3, n, q_{ID}, q_D)$-CCA-secure in the sID security model for $t_3 < t_1 - \Theta(\tau bn)$ and $\epsilon_1 + \epsilon_2 \ge \epsilon_3$, where τ is the maximum time for an exponentiation in $\mathbb{G}$.*

PROOF. Suppose there exists an adversary $\mathcal{A}$ which has advantage ϵ_3 in attacking the CCA security of the mIB-KEM. We construct an algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to solve the decision $(b+2)$-BDHE problem in $\mathbb{G}$. For a generator $g \in \mathbb{G}$ and $\alpha \in \mathbb{Z}_p$, let $g_i = g^{(\alpha^i)} \in \mathbb{G}$. On input $(z, g, g_1, \ldots, g_{b+1}, g_{b+3}, \ldots, g_{2b+2}, T)$, $\mathcal{B}$ outputs 1 if $T = e(z, g_{b+2})$ and 0 otherwise. $\mathcal{B}$ works by interacting with $\mathcal{A}$ as follows:

Init: $\mathcal{A}$ outputs a set S^* that it intends to attack.

Setup: As before, $\mathcal{B}$ first divides the challenge set S^* into subsets $\mathsf{S}^*_1, \ldots, \mathsf{S}^*_a$ after selecting two publicly computable surjective functions $H_1 : \mathbb{Z}_p^* \to \{1, \ldots, a\}$ and $H_2 : \mathbb{Z}_p^* \to \{1, \ldots, b\}$. Next, $\mathcal{B}$ computes $\mu^* = H_k(z)$ and selects two random $\rho, \tau \in \mathbb{Z}_p$. It sets $x_0 = g_{b+1} \cdot g^{\rho}$ and $h = g_{b+1} \cdot g^{\tau}$. Next, it picks random $\gamma_1, \ldots, \gamma_b, \delta_1, \ldots, \delta_a \in \mathbb{Z}_p$. It sets $y_i = g^{\gamma_i} g_i$ for $i = 1, \ldots, b$ and sets $x_j = g^{\delta_j} \cdot (\Pi_{\mathsf{ID} \in \mathsf{S}^*_j} (g_{H_2(\mathsf{ID})})^{\mathsf{ID}})^{-1} \cdot g_{b+1}^{-\mu^*}$ for $j = 1, \ldots, a$. It additionally picks a random hash key $k \in \mathcal{K}$ for hash function H. With the information about $(\mathbb{G}, \mathbb{G}_T, e, p, H_k)$, $\mathcal{B}$ gives $\mathcal{A}$ the public parameters

$$\mathsf{PP} = (g, g_1, h, x_0, x_1, \ldots, x_a, y_1, \ldots, y_b, H_1, H_2).$$

Since $\rho, \tau, \{\gamma_i\}$, and $\{\delta_j\}$ values are chosen uniformly at random, this public parameters have an identical distribution to that in the actual construction.

Phase 1: $\mathcal{A}$ issues up to q_{ID} private key and q_D decryption queries. First, $\mathcal{B}$ needs to generate private keys d_{ID} for $\mathsf{ID} \notin \mathsf{S}^*$. Consider a private key for ID such that $H_1(\mathsf{ID}) = u$ and $H_2(\mathsf{ID}) = v$. As before, we can assume the pair (u, v) does not belong to the set $\{(H_1(\mathsf{ID}), H_2(\mathsf{ID})) : \mathsf{ID} \in \mathsf{S}^*\}$. $\mathcal{B}$ picks a random $r \in \mathbb{Z}_p$. Let $r' = r - \alpha^{(b+2-v)}/\ \mathsf{ID}$. ($\mathsf{ID} \in \mathbb{Z}_p^*$ as before.) $\mathcal{B}$ generates the private key d_{ID} for ID as

$$(\ x_0^{\alpha} \cdot (x_u \cdot y_v^{\mathsf{ID}})^{r'},\ h^{r'},\ g^{r'},\ y_1^{r'}, \ldots,\ y_{v-1}^{r'}, \ldots,\ y_{v+1}^{r'}, \ldots,\ y_b^{r'}\ )$$

which is a properly distributed private key for ID. By the similar calculation to that in Section 3, we can show that $\mathcal{B}$ is able to compute all elements of this private key given the input values, except $h^{r'}$. The term $h^{r'}$ becomes

$$h^{r'} = (g_{b+1} \cdot g^{\tau})^{r} \cdot (g_{2b+3-v} \cdot g_{b+2-v}^{\tau})^{-1/\mathsf{ID}}.$$

Since $1 \le v \le b$, the unknown value g_{b+2} is not required to compute $h^{r'}$.

Second, let $(\mathsf{ID}, \mathsf{S}, \mathsf{Hdr})$ be a decryption query where $\mathsf{S} \subseteq \mathsf{S}^*$ and $\mathsf{ID} \in \mathsf{S}$. Let $\mathsf{Hdr} = (A_1, \ldots, A_a, B)$. Wlog, let $H_1(\mathsf{ID}) = u$ and $H_2(\mathsf{ID}) = v$ for ID. When we divide S into subsets $(\mathsf{S}_1, \ldots, \mathsf{S}_a)$, we have that $\mathsf{ID} \in \mathsf{S}_u \subseteq \mathsf{S}^*_u$. To decrypt the queried ciphertext, $\mathcal{B}$ does as follows:

1. Compute $\mu' = H_k(B)$ and check if the components (A_u, B) in the Hdr are of the valid form, using the following equation

   $$e(A_u,\ g) = e(x_u \cdot h^{\mu'} \cdot \Pi_{\mathsf{ID} \in \mathsf{S}_u} (y_{H_2(\mathsf{ID})})^{\mathsf{ID}},\ B).$$

   If the equality does not hold, $\mathcal{B}$ responds with $\perp$.

2. Otherwise, check that $\mu' = \mu^*$. If the equality holds, $\mathcal{B}$ outputs a random bit $b \in \{0, 1\}$ and aborts the simulation (in this case, the collision of hash function H_k occurs).

3. Otherwise, from the equation above, $\mathcal{B}$ has that

   $$A_u = (x_u \cdot h^{\mu'} \cdot \Pi_{\mathsf{ID} \in \mathsf{S}_u} (y_{H_2(\mathsf{ID})})^{\mathsf{ID}})^{s}$$

   for some (unknown) $s \in \mathbb{Z}_p$ such that $B = g^s$. Plugging in the values of x_u, h, and $\{y_k\}$, the A_u becomes

   $$\begin{aligned} A_u &= (g^{\delta_u} (\Pi_{\mathsf{ID} \in \mathsf{S}^*_u} (g_{H_2(\mathsf{ID})})^{-\mathsf{ID}})^{-1} g_{b+1}^{-\mu^*} \cdot (g_{b+1} g^{\tau})^{\mu'} \cdot \\ &\qquad \Pi_{\mathsf{ID} \in \mathsf{S}_u} (g^{\gamma_{H_2(\mathsf{ID})}} g_{H_2(\mathsf{ID})})^{\mathsf{ID}})^{s} \\ &= (g_{b+1}^{(\mu' - \mu^*)} \cdot g^{\eta} \cdot \Pi_{\mathsf{ID} \in \mathsf{S}^*_u \setminus \mathsf{S}_u} (g_{H_2(\mathsf{ID})})^{-\mathsf{ID}})^{s} \end{aligned}$$

   where $\eta = \delta_u + \tau\mu' + \Sigma_{\mathsf{ID} \in \mathsf{S}_u} \mathsf{ID} \cdot \gamma_{H_2(\mathsf{ID})}$. $\mathcal{B}$ computes

   $$\begin{aligned} d'_{\mathsf{ID},1} &= g_1^{-\eta/(\mu' - \mu^*)} \cdot A_u \cdot \\ &\qquad (\Pi_{\mathsf{ID} \in \mathsf{S}^*_u \setminus \mathsf{S}_u} (g_{H_2(\mathsf{ID})+1})^{\mathsf{ID}})^{-1/(\mu' - \mu^*)}, \\ d'_{\mathsf{ID},3} &= B \cdot g_1^{-1/(\mu' - \mu^*)}. \end{aligned}$$

   Since $1 \le H_2(\mathsf{ID}) \le b$, $\mathcal{B}$ does not require knowledge of g_{b+2} and then is able to compute $d'_{\mathsf{ID},1}$ with input values. Let $r' = s - \alpha/(\mu' - \mu^*)$. Then,

   $$\begin{aligned} d'_{\mathsf{ID},1} &= g_1^{-\eta/(\mu' - \mu^*)} (g_{b+1}^{(\mu' - \mu^*)} g^{\eta} \cdot \Pi_{\mathsf{ID} \in \mathsf{S}^*_u \setminus \mathsf{S}_u} (g_{H_2(\mathsf{ID})})^{-\mathsf{ID}})^{s} \\ &\qquad \cdot (\Pi_{\mathsf{ID} \in \mathsf{S}^*_u \setminus \mathsf{S}_u} (g_{H_2(\mathsf{ID})+1})^{\mathsf{ID}})^{-1/(\mu' - \mu^*)} \\ &= g_{b+2} \cdot (g_{b+1}^{(\mu' - \mu^*)} \cdot g^{\eta} \cdot \Pi_{\mathsf{ID} \in \mathsf{S}^*_u \setminus \mathsf{S}_u} (g_{H_2(\mathsf{ID})})^{-\mathsf{ID}})^{r'} \\ &= g_{b+2} \cdot (x_u \cdot h^{\mu'} \cdot \Pi_{\mathsf{ID} \in \mathsf{S}_u} (y_{H_2(\mathsf{ID})})^{\mathsf{ID}})^{r'}, \\ d'_{\mathsf{ID},3} &= g^{s} g_1^{-1/(\mu' - \mu^*)} = g^{r'}. \end{aligned}$$

   Recall that $x_0^{\alpha} = g_{b+2} \cdot g_1^{\rho}$. For the re-randomization, $\mathcal{B}$ selects a random $r'' \in \mathbb{Z}_p$ and computes $d''_{\mathsf{ID},1} = d'_{\mathsf{ID},1} \cdot g_1^{\rho} \cdot (x_u \cdot h^{\mu'} \cdot \Pi_{\mathsf{ID} \in \mathsf{S}_u} (y_{H_2(\mathsf{ID})})^{\mathsf{ID}})^{r''}$ and $d''_{\mathsf{ID},3} = d'_{\mathsf{ID},3} \cdot g^{r''}$. For some (unknown) $r''' = r' + r''$,

   $$\begin{aligned} d''_{\mathsf{ID},1} &= x_0^{\alpha} \cdot (x_u \cdot h^{\mu'} \cdot \Pi_{\mathsf{ID} \in \mathsf{S}_u} (y_{H_2(\mathsf{ID})})^{\mathsf{ID}})^{r'''}, \\ d''_{\mathsf{ID},3} &= g^{r'''}. \end{aligned}$$

   $\mathcal{B}$ responds with $e(d''_{\mathsf{ID},1}, B)/e(A_u, d''_{\mathsf{ID},3})$. This response is identical to the decapsulation algorithm in a real attack, because r'' (and r''') is uniform in $\mathbb{Z}_p$.

Challenge: $\mathcal{B}$ computes Hdr^* as

$$(z^{\delta_1 + \tau\mu^* + \Sigma_{\mathsf{ID} \in \mathsf{S}^*_1} \mathsf{ID} \cdot \gamma_{H_2(\mathsf{ID})}},\ \ldots,\ z^{\delta_a + \tau\mu^* + \Sigma_{\mathsf{ID} \in \mathsf{S}^*_a} \mathsf{ID} \cdot \gamma_{H_2(\mathsf{ID})}},\ z)$$

and $K^* = T \cdot e(g_1, z^{\rho})$, where z and T are input values given to $\mathcal{B}$. Recall that $\mu^* = H_k(z)$. As before, if $z = g^c$ for some (unknown) $c \in \mathbb{Z}_p$, then

$$\begin{aligned} & z^{\delta_i + \tau\mu^* + \Sigma_{\mathsf{ID} \in \mathsf{S}^*_i} \mathsf{ID} \cdot \gamma_{H_2(\mathsf{ID})}} \\ &= (g^{\delta_i} \cdot (\Pi_{\mathsf{ID} \in \mathsf{S}^*_i} (g_{H_2(\mathsf{ID})})^{\mathsf{ID}})^{-1} \cdot g_{b+1}^{-\mu^*} \cdot (g_{b+1} g^{\tau})^{\mu^*} \cdot \\ &\qquad \Pi_{\mathsf{ID} \in \mathsf{S}^*_i} (g^{\gamma_{H_2(\mathsf{ID})}} g_{H_2(\mathsf{ID})})^{\mathsf{ID}})^{c} \\ &= (x_i \cdot h^{\mu^*} \cdot \Pi_{\mathsf{ID} \in \mathsf{S}^*_i} (y_{H_2(\mathsf{ID})})^{\mathsf{ID}})^{c} \end{aligned}$$

for $i = 1, \ldots, a$. If $T = e(z, g_{b+2})$ then $K^* = e(x_0, g_1)^c$ and thus (Hdr^*, K^*) is a valid challenge to $\mathcal{A}$ for the receiver set S^*. On the other hand, when T is uniform and independent in $\mathbb{G}_T$, then Hdr^* is independent of K^* in the adversary's view.

Phase 2: $\mathcal{A}$ issues more private key and decryption queries not queried in phase 1. $\mathcal{B}$ responds as before.

Guess: $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$. If $b = b'$ then $\mathcal{B}$ outputs 1, indicating $T = e(z, g_{b+2})$. Otherwise, it outputs 0, indicating $T \neq e(z, g_{b+2})$.

When T is random in $\mathbb{G}_T$ then $\Pr[\mathcal{B}(z, g, \overrightarrow{g}_{\alpha,b+1}, T) = 0] = 1/2$. Let $\mathsf{Collision}$ denote the event that $\mathcal{A}$ submits a valid header $\mathsf{Hdr} = (A_1, \ldots, A_a, B)$ such that $\mu^* = H_k(B)$ as a decapsulation query. In the case of $\mathsf{Collision}$, $\mathcal{B}$ cannot reply to the decryption query and aborts the simulation. When $T = e(z, g_{b+2})$, $\mathcal{B}$ replied with a valid message encryption key unless event $\mathsf{Collision}$ occurs. Then, $\mathcal{B}$ has

$$|\Pr[\mathcal{B}(z, g, \overrightarrow{g}_{\alpha,b+1}, T) = 0] - 1/2| \geq$$
$$|\Pr[b = b' \wedge \overline{\mathsf{Collision}}] - 1/2| - \Pr[\mathsf{Collision}].$$

Since $\mathcal{B}$ provided $\mathcal{A}$ with perfect simulation when the event $\mathsf{Collision}$ did not occur, $|\Pr[b = b' \wedge \overline{\mathsf{Collision}}] - 1/2| \geq \epsilon_3$. Also, note that $\Pr[\mathsf{Collision}]$ is negligible. This means that $\Pr[\mathsf{Collision}] < \epsilon_2$ since otherwise $\mathcal{B}$ finds two values z, B such that $H_k(z) = H_k(B)$, which is contradiction to the definition of H. Therefore,

$$|\Pr[\mathcal{B}(z, g, \overrightarrow{g}_{\alpha,b+1}, e(z, g_{b+2})) = 0] -$$
$$\Pr[\mathcal{B}(z, g, \overrightarrow{g}_{\alpha,b+1}, T) = 0]| \geq \epsilon_3 - \epsilon_2.$$

This completes the proof of Theorem 2. □

6. CONCLUSION

We showed that a mIB-KEM suggested by Chatterjee and Sarkar [12] does not guarantee the chosen plaintext (or ciphertext) security. The security leak is originated from embedding the same element into partitioned subsets of receivers. We solved this weakness to use random elements dedicated to subsets respectively. Our proposed mIB-KEM has sublinear-size ciphertexts and private keys. We proved the chosen plaintext security without random oracles under the BDHE assumption, and extended the CPA-secure scheme to obtain the chosen ciphertext security by employing the hash-based transformation [9].

Acknowledgments

We would like to thank the anonymous referees for their helpful comments.

7. REFERENCES

[1] M. Abdalla, E. Kiltz, and G. Neven. Generalized key delegation for hierarchical identity-based encryption. In *Proc. ESORICS 2007*, volume 4734 of LNCS, pages 139–154. Springer, 2007.

[2] J. Baek, R. Safavi-Naini, and W. Susilo. Efficient multi-receiver identity-based encryption and its application to broadcast encryption. In *Proc. PKC 2005*, volume 3386 of LNCS, pages 380–397. Springer, 2005.

[3] M. Barosa and P. Farshim. Efficient identity-based key encapsulation to multiple parties. In *Proc. IMA 2005*, volume 3796 of LNCS, pages 428–441. Springer, 2005.

[4] D. Boneh and X. Boyen. Efficient selective-id secure identity based encryption without random oracles. In *Proc. Eurocrypt 2004*, volume 3027 of LNCS, pages 223–238. Springer, 2004.

[5] D. Boneh, X. Boyen, and E. Goh. Hierarchical identity based encryption with constant size ciphertext. In *Proc. Eurocrypt 2005*, volume 3494 of LNCS, pages 440–456. Springer, 2005.

[6] D. Boneh and M. Franklin. Identity-based encryption from the weil pairing. In *Proc. CRYPTO 2001*, volume 2139 of LNCS, pages 213–229. Springer, 2001.

[7] D. Boneh, C. Gentry, and B. Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In *Proc. CRYPTO 2005*, volume 3621 of LNCS, pages 258–275. Springer, 2005.

[8] D. Boneh and J. Katz. Improved efficiency for cca-secure cryptosystems built using identity-based encryption. In *Proc. CT-RSA 2005*, volume 3376 of LNCS, pages 87–103. Springer, 2005.

[9] X. Boyen, Q. Mei, and B. Waters. Direct chosen ciphertext security from identity-based techniques. In *ACM Conference on Computer and Communications Security - CCS 2005*, pages 320–329. New-York: ACM Press, 2005.

[10] C. Canetti, S. Halevi, and J. Katz. Chosen ciphertext security from identity-based encryption. In *Proc. Eurocrypt 2004*, volume 3027 of LNCS, pages 207–222. Springer, 2004.

[11] S. Chatterjee and P. Sarkar. Generalization of the selective-id security model for hibe protocols. In *Proc. PKC 2006*, volume 3958 of LNCS, pages 241–256. Springer, 2006.

[12] S. Chatterjee and P. Sarkar. Multi-receiver identity-based key encapsulation with shortened ciphertext. In *Proc. Indocrypt 2006*, volume 4329 of LNCS, pages 394–408. Springer, 2006.

[13] C. Cocks. ibe–3. In *TeX90 Conference Proceedings*, pages 84–89. TeX Users Group, March 1991.

[14] X. Du, Y. Wang, J. Ge, and Y. Wang. An id-based broadcast encryptoin scheme for key distribution. *IEEE Transaction on Broadcasting*, 51(2):264–266, 2005.

[15] D. Galindo and E. Kiltz. Direct chosen ciphertext secure identity-based key encapsulation without random oracles. In *Proc. ASISP 2006*, volume 4058 of LNCS, pages 336–347. Springer, 2006.

[16] C. Gentry. Practical identity-based encryption without random oracles. In *Proc. Eurocrypt 2006*, volume 4004 of LNCS, pages 445–464. Springer, 2006.

[17] C. Gentry and A. Silverberg. Hierarchical id-based cryptography. In *Proc. Asiacrypt 2002*, volume 2501 of LNCS, pages 548–566. Springer, 2002.

[18] R. Sakai and J. Furukawa. Identity-based broadcast encryption. In *Cryptology ePrint Archive.* Report 2007/217, http://eprint.iacr.org/2007/217, 2007.

[19] A. Shamir. Identity-based cryptosystems and signature schemes. In *Proc. CRYPTO 1984*, volume 196 of LNCS, pages 47–53. Springer, 1984.

[20] B. Waters. Efficient identity-based encryption without random oracles. In *Proc. Eurocrypt 2005*, volume 3494 of LNCS, pages 114–124. Springer, 2005.