ISSN 1342-2812

Research Reports on Mathematical and Computing Sciences

An ID-based Combined Scheme with Encryption and Signature

Naoyuki Yamashita and Keisuke Tanaka

December 2006, C–231

Department of
Mathematical and
Computing Sciences
Tokyo Institute of Technology

SERIES C: Computer Science

An ID-based Combined Scheme with Encryption and Signature

Naoyuki Yamashita and Keisuke Tanaka*

Dept. of Mathematical and Computing Sciences
Tokyo Institute of Technology
W8-55, 2-12-1 Ookayama Meguro-ku, Tokyo 152-8552, Japan
{yamashi1, keisuke}@is.titech.ac.jp

December 27, 2006

Abstract

The cryptographic protocol known as the combined scheme allows users to decrypt ciphertexts and create signatures using the same key. In this paper, we model ID-based combined schemes. Many ID-based encryption schemes and ID-based signature schemes are proposed. Most of them are based on bilinear maps. Although it seems possible to combine these schemes, there is no security definition for the combination. We propose a model for this combinations, and define the security condition.

As an additional property, a definition for the key privacy of encryption schemes is proposed. In the combined scheme, the same private key is used for the decryption and the signing. To protect the owner's privacy, both the encryption scheme and the signature scheme must satisfy the key privacy condition. When combining them, the encryption scheme should not degrade the key privacy of the signature scheme, and vice versa. We propose a key privacy condition for ID-based signature schemes. We then modify this notion to ID-based combined schemes. We construct a concrete scheme and prove that this scheme satisfies these security requirements.

Furthermore, we discuss several related topics. First, we propose a definition for the key privacy of a non-ID-based combined scheme, and prove that the prior scheme satisfies this property with a short modification. We also propose an ID-based combined scheme with multiple-receiver encryption scheme. We prove the security of this concrete scheme.

Keywords: Combined scheme, ID-based encryption, ID-based signature, key privacy.

1 Introduction

Identity based scheme

In 1984, Shamir [26] asked for identity (ID)-based encryption and signature schemes to simplify key management procedures of certificate-based public key infrastructures. Since then, several ID-based encryption schemes and signature schemes have been proposed. In 2001, Boneh and Franklin [9] proposed a practical ID-based encryption scheme. Their scheme was based on bilinear maps, and many schemes based on this scheme were then proposed.

Baek, Safavi-Naini, and Susilo [2] proposed a multi-receiver ID-based encryption scheme. This scheme is more efficient than re-encrypting a message n times using Boneh and Franklin's scheme. The ID-based signcryption scheme [3, 24] creates a ciphertext which includes a signature of the plaintext. Boyen's ID-based signcryption scheme [11] is as compact and as efficient as taking each

*Supported in part by NTT Information Sharing Platform Laboratories and Grant-in-Aid for Scientific Research, Ministry of Education, Culture, Sports, Science, and Technology, 16092206.

scheme separately. Smart [27] proposed an authenticated key agreement scheme. The notion of the ID-based encryption scheme was modified to the hierarchical ID-based encryption by Gentry and Silverberg [20], and Horwits and Lynn [23]. In an ID-based encryption scheme, the public key generator has to generate a private key for every identity. In a hierarchical ID-based encryption scheme, this workload is reduced by delegating the private key generation task to lower level entities.

One open problem that Boneh and Franklin [9] asked is the construction of an ID-based encryption scheme without random oracles. Boneh and Boyen [6] proposed a scheme which satisfies a weaker security notion without random oracles. This weaker security notion was proposed by Canetti, Halevi, and Kats [14]. In this model, an attacker chooses an identity which he attacks before it activated. Later Boneh and Boyen [7] modified the prior scheme to satisfy the full identity security without random oracles. In this model, an attacker can choose the target identity in a challenge phase. This notion is the security which was proposed originally. Independently from Boneh and Boyen [7], Waters [28] constructed a scheme which is secure without random oracles. Gentry [18] proposed a scheme which is also secure without random oracles, and is more efficient than these schemes. Canetti et al. [14] also proposed a conversion from a weaker security ID-based encryption scheme to a CCA-secure public key encryption scheme. Since this conversion does not change the model, it could be used to construct a CCA-secure public key encryption scheme without random oracles.

In addition to their obvious privacy benefits, there has been interest in the anonymity of ID-based encryption schemes, which is also called key privacy. This property was first observed by Boneh, Crescenzo, Ostrovsky, and Persiano [8], and later formalized by Abdalla, Bellare, Catalano, Kiltz, Khono, Lange, Malone-Lee, Neven, Paillier, and Shi [1]. Boyen [11] noticed the anonymity of Boneh and Franklin's scheme. An anonymous hierarchical ID-based encryption scheme was proposed by Boyen and Waters [12]. Their scheme is the first hierarchical ID-based scheme whose anonymity is proven without random oracles.

Paterson [25], Hess [22], and Cha and Cheon [15] proposed ID-based signature schemes. In an aggregate signature scheme [10], multiple signatures can be aggregated into a compact aggregate signature, even if these signatures are on different documents and were produced by different signers. Gentry and Ramzan [19] proposed an identity-based aggregate signature scheme. In this scheme, the verifier needs only a description of who signed what and the single public key of a private key generator. Bellare, Neven, and Namprempre [5] demonstrated that ID-based signature schemes can be constructed from any PKI-based signature scheme. Galindo, Herranz, and Kiltz [17] proposed a generic conversion from any public key signature scheme to an ID-based signature scheme which preserves its additional property.

Combined scheme

Public key systems supporting both encryption and signature generation with a single private key per user model a public key in a constrained environment that can afford only a single key per user. These systems are combinations of encryption and signature schemes, so they are called combined schemes. The user is required both to sign and to decrypt with this single secret key. This constraint may help an attack on the combined scheme to obtain more information than an individual attack on each scheme separately. An attack on the encryption scheme may use some information obtained from the signature scheme, and an attack on the signature scheme may use some information obtained from the encryption scheme.

Haber and Pinkas [21] analyzed the security of some combination of some encryption schemes and some signature schemes. Diament, Lee, Keromytis, and Yung [16] modeled the combined scheme and defined the security for this model. They constructed a concrete scheme, and proved the security.

Key privacy for signatures

Public key encryption provides data confidentiality to many kinds of applications. Bellare, Boldyreva, Desai, and Pointcheval [4] introduced public key encryption with key privacy and formalized it.

On signer privacy, however, it is generally unclear and doubtful whether a signature scheme alone can guarantee the anonymity of the signer. Several techniques to convert an encryption scheme to a key privacy enabled version were proposed by Bellare et al. [4], however, these techniques cannot be simply applied to digital signature schemes to convert them to an anonymous version. Yang, Wong, Deng, and Wang [29] proposed the first formal definition of the signer anonymity for digital signatures and constructed concrete schemes.

Our contribution

In this paper, we propose an ID-based combined scheme. Many ID-based encryption schemes and ID-based signature schemes exist. Most of them are based on bilinear maps. Although it seems possible to combine these schemes, there are no security definition for the combination. We propose the first definition for it, which is a modification of the security definition of the non-ID-based combined scheme proposed by Diament et al. [16].

Secondly, we propose a definition for the key privacy of an ID-based signature scheme. Similarly to the non-identity based signature scheme, we define key privacy as the impossibility of an attacker to identify the key. For the choice of the target IDs, both adaptive and selective games can be defined. In an adaptive game, the attacker can choose the target ID when it outputs the ID with a message and a forged signature. In a selective game, the attacker fixes the target ID before it interacts with the oracles. In this paper, we adopt the adaptive game and show that the signature scheme by Cha and Cheon [15] satisfies this definition.

Furthermore, we propose a key privacy definition for the ID-based combined scheme. The combined scheme allows users to decrypt ciphertexts and create signatures using the same key. To protect the owner's privacy, both the encryption scheme and the signature scheme must satisfy the key privacy condition. When combining them, the encryption scheme should not degrade the key privacy of the signature scheme, and vice versa. Intuitively, the adversary should not be able to distinguish the key even if he is allowed to access the decryption oracle and signature oracle and even if he receives both a ciphertext and a signature as challenges. We define the key privacy when no attacker can distinguish between two IDs. We construct a concrete scheme combined the encryption scheme of Boneh and Franklin [9] with the signature scheme by Cha and Cheon [15]. Cha and Cheon insisted that their scheme can share the parameters, but no security argument was given. We prove that this scheme satisfies these security properties.

Organization

The organization of this paper is as follows. In section 2, we define the computational assumptions on which the security of concrete schemes is based. In section 3, we modify the notion of key privacy to ID-based signature schemes, and prove the key privacy for the signature scheme by Cha and Cheon [15]. In section 4, we model the ID-based combined scheme, construct a concrete scheme, and prove its security. This security definition includes the key privacy for ID-based combined schemes. In section 5, we discuss the key privacy for combined schemes. In section 6, we prove the security of a particular combination, which is composed from an ID-based combined scheme with multiple-receivers and the ID-based signature scheme by Cha and Cheon [15].

2 Computational assumptions

In this section, we review the bilinear maps and the related computational assumptions on which the security of our concrete schemes are based.

Definition 1 (The discrete logarithm problem)**.** *Let $\mathcal{G}$ be a group of prime order q. Let P be a generator of $\mathcal{G}$, and a an element of $\mathbb{Z}_q$. The discrete logarithm (DL) problem is given (P, aP), to compute a.*

We say that the DL problem is hard when there is no probabilistic polynomial-time attacker A that solves the DL problem with non-negligible probability.

Definition 2 (The computational Diffie-Hellman problem)**.** *Let $\mathcal{G}$ be a group of prime order q. Let P be a generator of $\mathcal{G}$, and a and b elements of $\mathbb{Z}_q$. The computational Diffie-Hellman (CDH) problem is given (P, aP, bP), to compute abP.*

We say that the CDH problem is hard when there is no probabilistic polynomial-time attacker A that solves the CDH problem with non-negligible probability.

Definition 3 (The decisional Diffie-Hellman problem)**.** *Let $\mathcal{G}$ be a group of prime order q. Let P be a generator of $\mathcal{G}$, and a, b, and c elements of $\mathbb{Z}_q$. The decisional Diffie-Hellman (DDH) problem is given (P, aP, bP, cP), to decide whether $ab \equiv c \bmod q$.*

We say that the DDH problem is hard when there is no probabilistic polynomial-time attacker A that solves the DDH problem with a non-negligible probability strictly greater than $1/2$. We call a tuple (P, aP, bP, cP) valid if $ab \equiv c \bmod q$.

Definition 4 (The gap Diffie-Hellman problem)**.** *Let $\mathcal{G}$ be a group of prime order q. Let P be a generator of $\mathcal{G}$. Let a, b, and c be elements of $\mathbb{Z}_q$. The gap Diffie-Hellman (GDH) problem is given (P, aP, bP), to solve the CDH problem with the help of a DDH oracle that is able to decide whether or not a tuple $(P, a'P, b'P, c'P)$ is valid.*

We say that the GDH problem is hard when there is no probabilistic polynomial-time attacker A that solves the GDH problem with non-negligible probability.

Definition 5 (Bilinear maps)**.** *Let $\mathcal{G}, \mathcal{F}$ be groups of the same prime order q. A function $\hat{e} : \mathcal{G} \times \mathcal{G} \to \mathcal{F}$ with the following three properties is a bilinear map.* `(i)` *Bilinear: For any $P_1, P_2 \in \mathcal{G}$, and $a, b \in \mathbb{Z}_q$, $\hat{e}(aP_1, bP_2) = \hat{e}(P_1, P_2)^{ab}$.* `(ii)` *Non-degenerate: If P is a generator of $\mathcal{G}$, then $\hat{e}$ does not map (P, P) to the unit element in $\mathcal{F}$.* `(iii)` *Computable: For all $P_1, P_2 \in \mathcal{G}$, the map $\hat{e}(P_1, P_2)$ is efficiently computable.*

According to the property `(ii)`, if P is a generator of $\mathcal{G}$, then $\hat{e}(P, P)$ is a generator of $\mathcal{F}$.

We review the problem associated with the bilinear maps.

Definition 6 (The computational bilinear Diffie-Hellman problem)**.** *Let $\mathcal{G}, \mathcal{F}$ be groups of the same prime order q. Let P be a generator of $\mathcal{G}$. Let $\hat{e} : \mathcal{G} \times \mathcal{G} \to \mathcal{F}$ be a bilinear map. Let a, b, and c be elements of $\mathbb{Z}_q$. The computational bilinear Diffie-Hellman (CBDH) problem is given (P, aP, bP, cP), to compute $\hat{e}(P, P)^{abc}$.*

We say that the CBDH problem is hard when there is no probabilistic polynomial-time attacker A that solves the CBDH problem with non-negligible probability.

Definition 7 (The decisional bilinear Diffie-Hellman problem)**.** *Let $\mathcal{G}, \mathcal{F}$ be groups of the same prime order q. Let P be a generator of $\mathcal{G}$. Let $\hat{e} : \mathcal{G} \times \mathcal{G} \to \mathcal{F}$ be a bilinear map. Let a, b, and c be elements of $\mathbb{Z}_q$, and h an element of $\mathcal{F}$. The decisional bilinear Diffie-Hellman (DBDH) problem is given (P, aP, bP, cP, h), to decide whether or not $\hat{e}(P, P)^{abc} = h$.*

We say that the DBDH problem is hard when there is no probabilistic polynomial-time attacker A that solves the DBDH problem with a non-negligible probability strictly greater than $1/2$. We call a tuple (P, aP, bP, cP, h) valid if $\hat{e}(P, P)^{abc} = h$.

Definition 8 (The gap bilinear Diffie-Hellman problem)**.** *Let $\mathcal{G}, \mathcal{F}$ be groups of the same prime order q. Let P be a generator of $\mathcal{G}$. Let $\hat{e} : \mathcal{G} \times \mathcal{G} \to \mathcal{F}$ be a bilinear map. Let a, b, and c be elements of $\mathbb{Z}_q$. The gap bilinear Diffie-Hellman (GBDH) problem is given (P, aP, bP, cP), to solve the CBDH problem with the help of a DBDH oracle that is able to decide whether or not a tuple $(P, a'P, b'P, c'P, h)$ is a valid DBDH tuple.*

We say that the GBDH problem is hard when there is no probabilistic polynomial-time attacker A that solves the GBDH problem with non-negligible probability.

3 Key privacy for ID-based signature schemes

In this section, we define the key privacy for ID-based signature schemes. This notion comes from the definition of the key privacy for encryption schemes [29]. We prove that the ID-based signature scheme by Cha and Cheon [15] satisfies this definition.

3.1 Model for ID-based signature schemes

First, we define the model for the ID-based signature scheme. In this model, there are a private key generator (PKG) and some users. Each user has an identity ID and a corresponding private key obtained from the private key generator.

Definition 9. *An ID-based signature scheme consists of four algorithms* `IDSig` *=* (KeyGen, Extract, Sign, Verify) *as follows.*

- *The private key generator's key generation algorithm* KeyGen*: The private key generator runs this algorithm to generate a private key generator's master key and a common parameter, denoted by mk_{PKG} and cp_{PKG}, respectively. Note that cp_{PKG} is published publicly while mk_{PKG} is kept secret.*
- *The private key generator's private key extraction algorithm* Extract*: Given an identity ID received from a user and the private key generator's master key mk_{PKG} as input, the private key generator runs this algorithm to generate a private key associated with ID, denoted by sk_{ID}. We denote $sk_{ID} = \mathsf{Extract}(mk_{PKG}, ID)$.*
- *The signature algorithm* Sign*: Given the signer's private key sk_{ID}, the private key generator's common parameter cp_{PKG}, and a message M as input, the signer runs this algorithm to create a signature σ. We denote $\sigma = \mathsf{Sign}(cp_{PKG}, sk_{ID}, M)$.*
- *The verification algorithm* Verify*: Receiving a message M, a signature σ, and an identity ID, the receiver checks the validity of the signature. If it is a valid tuple, it produces an accept symbol* accept*. Otherwise, it produces a reject symbol* reject*. We denote $\mathsf{Verify}(cp_{PKG}, ID, M, \sigma) =$ accept or* reject*.*

In a standard public key encryption scheme, the capability to recover plaintexts from ciphertexts is due to the secret key. A standard encryption scheme is said to respect key privacy if the ciphertexts do not reveal their ownership. Which means that the attacker cannot discover the ownership for fixed public keys even if he possesses all the information except for the secret keys. However, in an ID-based encryption scheme, this can be done not only with the knowledge of the secret key but also the master key. Using the master key, the attacker can compute the secret key for any identity. Thus, the security condition should ensure the secrecy of the master key. The key privacy is defined as the incapability of an attacker to identify the ownership of a ciphertext for any identity without the knowledge of the corresponding secret keys.

We adapt this notion to the key privacy for ID-based signature schemes. Yang et al. [29] defined the key privacy for standard signature schemes as the incapability of an attacker to discover the

generator of a signature. This notion can be modified for the ID-based signature scheme. We define the key privacy for ID-based signature schemes in Definition 10. Intuitively, the adversary should not be able to distinguish between two IDs in the following game even if he is allowed to access the extraction oracle and the signature oracle.

Definition 10. *The ID-based signature scheme* `IDSig` *is* ID-IK-CMA secure *if no PPT adversary* $\mathcal{A}$ *has a non-negligible advantage against a challenger in the following game.*

1. *The adversary* $\mathcal{A}$ *is given the common parameter* cp_{PKG}*.*
2. $\mathcal{A}$ *is allowed to query the signature oracle and the extraction oracle adaptively. The signature oracle receives an identity* ID *and a message* M *and returns the signature* $\sigma = \mathsf{Sign}(cp_{PKG}, sk_{ID}, M)$*. The extraction oracle receives an identity* ID *and returns the corresponding secret key* S_{ID}*.*
3. *When* $\mathcal{A}$ *announces to the challenger that he is ready, he outputs two target identities,* ID_0 *and* ID_1*. The challenger then tosses a random coin* $t \in \{0,1\}$*, chooses a message* $M' \in \mathcal{M}$ *randomly, and creates the signature* $\sigma' = \mathsf{Sign}(cp_{PKG}, sk_{ID_t}, M')$*, which he sends to* $\mathcal{A}$*.*
4. $\mathcal{A}$ *continues querying to the signature oracle and the extraction oracle adaptively. He is not allowed to query either* ID_0 *or* ID_1 *from the extraction oracle.*
5. *At the end of the game,* $\mathcal{A}$ *outputs a bit* t'*.* $\mathcal{A}$ *wins if* $t' = t$*.*

$\mathcal{A}$*'s advantage is defined as* $\mathbf{Adv} = \big|\Pr[t' = 1|t = 1] - \Pr[t' = 1|t = 0]\big|$*. The probability is taken over the coin tosses of both* $\mathcal{A}$ *and the challenger, including the coin toss for* t*.*

3.2 Cha and Cheon's scheme

In this section, we review the ID-based signature scheme proposed by Cha and Cheon [15] and prove that this scheme satisfies the ID-IK-CMA security notion. We make a short modification for the latter argument. In the original scheme, all the computation is done within the group $\mathcal{G}$. The original security is based on the hardness of the GDH problem in $\mathcal{G}$. We involve a bilinear map $\hat{e} : \mathcal{G} \times \mathcal{G} \to \mathcal{F}$ and check whether $\hat{e}(aP, bP) = \hat{e}(P, Q)$ instead of solving the DDH problem (P, aP, bP, Q).

In this setting, there are a public key generator PKG and some users. This ID-based signature scheme `IDSig` consists of the following four algorithms (KeyGen, Extract, Sign, Verify).

- KeyGen: The algorithm chooses two groups $\mathcal{G} = \langle P \rangle$ and $\mathcal{F}$ of the same prime order q. It constructs a bilinear map $\hat{e} : \mathcal{G} \times \mathcal{G} \to \mathcal{F}$. It chooses $s \in \mathbb{Z}_q^*$ uniformly at random and computes $P_{pub} = sP$. Also, it selects hash functions $H_{\mathtt{ID}} : \{0,1\}^* \to \mathcal{G}$, $H_1 : \{0,1\}^* \times \mathcal{G} \to \mathbb{Z}_q$. The system parameter is $(P, P_{pub}, \hat{e}, H_{\mathtt{ID}}, H_1)$. The master key is s. We remark that these hash functions will be viewed as random oracles in our security proof.
- Extract: Given an identity ID, the algorithm computes $S_{ID} = sH_{\mathtt{ID}}(ID)$ and outputs it as the private key associated to ID. We remark that $H_{\mathtt{ID}}(ID)$ plays the role of the associated public key.
- Sign: Given a secret key S_{ID} and a message m, the algorithm picks a random number $r \in \mathbb{Z}_q$ and outputs a signature $\sigma = (U, V)$ where $U = rH_{\mathtt{ID}}(ID), V = (r + H_1(m, U))S_{ID}$.
- Verify: To verify a signature $\sigma = (U, V)$ of a message m for an identity ID, the algorithm checks whether $\hat{e}(P_{pub}, U + H_1(m, U)H_{\mathtt{ID}}(ID)) \stackrel{?}{=} \hat{e}(P, V)$. If this equation holds, it returns accept. Otherwise, it returns reject.

Theorem 1. *The ID-based signature scheme* IDSig *is ID-IK-CMA secure assuming the hardness of the DL problem for* $\mathcal{G}$ *in the random oracle model.*

The proof is shown in Appendix A.

4 ID-based combined scheme

In this section, we define the model for the ID-based combined scheme. Its security definition includes the notion of the key privacy of the ID-based combined scheme. We prove that an ID-based scheme which combines the encryption scheme by Boneh and Franklin [9] and the signature scheme by Cha and Cheon [15] satisfies these security properties.

4.1 Model for the ID-based combined scheme

Here we define the model for the ID-based combined scheme. In this model, there are a private key generator (PKG) and some users. Each user has an identity ID and a corresponding private key obtained from the private key generator.

Definition 11. *An ID-based combined scheme consists of six algorithms* IDComb $=$ *(*KeyGen*,* Extract*,* Encrypt*,* Decrypt*,* Sign*,* Verify*) as follows.*

- *The private key generator's key generation algorithm* KeyGen*: The private key generator runs this algorithm to generate a private key generator's master key and a common parameter, denoted by* mk_{PKG} *and* cp_{PKG}*, respectively. Note that* cp_{PKG} *is published publicly while* mk_{PKG} *is kept secret.*

- *The private key generator's private key extraction algorithm* Extract*: Given an identity* ID *received from a user and the private key generator's master key* mk_{PKG} *as input, the private key generator runs this algorithm to generate a private key associated with* ID*, denoted by* sk_{ID}*. We denote* $sk_{ID} = \mathsf{Extract}(mk_{PKG}, ID)$*.*

- *The encryption algorithm* Encrypt*: Given an identity* ID *of the receiver, the private key generator's common parameter* cp_{PKG}*, and a plaintext* M *as input, the sender runs this algorithm to generate a ciphertext* C *which is an encryption of* M *under* ID*. We denote* $C = \mathsf{Encrypt}(cp_{PKG}, ID, M)$*.*

- *The decryption algorithm* Decrypt*: Given the receiver's private key* sk_{ID}*, the private key generator's common parameter* cp_{PKG}*, and a ciphertext* C *as input, the receiver runs this algorithm to get a decryption* D*, which is either a certain plaintext or a reject symbol* reject*. We denote* $D = \mathsf{Decrypt}(cp_{PKG}, sk_{ID}, C)$*.*

- *The signature algorithm* Sign*: Given the signer's private key* sk_{ID}*, the private key generator's common parameter* cp_{PKG}*, and a message* M *as input, the signer runs this algorithm to create a signature* σ*. We denote* $\sigma = \mathsf{Sign}(cp_{PKG}, sk_{ID}, M)$*.*

- *The verification algorithm* Verify*: Receiving a message* M*, a signature* σ*, and an identity* ID*, the receiver checks the validity of the signature. If it is a valid tuple, it produces an accept symbol* accept*. Otherwise, it produces* reject*. We denote* $\mathsf{Verify}(cp_{PKG}, ID, M, \sigma) =$ accept *or* reject*.*

Here we define the security for the ID-based combined scheme. The user is required both to sign and to decrypt with a single secret key. This constraint may help an attack on the combined scheme to obtain more information than an individual attack on each scheme separately. An attack on the encryption scheme may use some information obtained from the signature scheme, and an attack on

the signature scheme may use some information obtained from the encryption scheme. Intuitively, we say the scheme is secure if there is no adversary which can neither distinguish ciphertexts nor forge a signature, even if it can choose which one to attack and access the extraction oracle, the decryption oracle, and the signature oracle.

Definition 12. *The ID-based combined scheme* `IDComb` *is* ID-IND-EUF secure *if no PPT adversary $\mathcal{A}$ has a non-negligible advantage against a challenger in the following joint ID-IND-EUF game.*

1. *The adversary $\mathcal{A}$ is given the common parameter cp_{PKG}.*
2. *$\mathcal{A}$ is allowed to query the decryption oracle, the signature oracle, and the extraction oracle adaptively. The decryption oracle receives an identity ID and a ciphertext C and returns the corresponding plaintext $D = \mathsf{Decrypt}(cp_{PKG}, sk_{ID}, C)$. The signature oracle receives an identity ID and a message M and returns the signature $\sigma = \mathsf{Sign}(cp_{PKG}, sk_{ID}, M)$. The extraction oracle receives an identity ID and returns the corresponding secret key S_{ID}.*
3. *When $\mathcal{A}$ announces to the challenger that he is ready, $\mathcal{A}$ chooses either an IND-CCA challenge or an EUF-CMA challenge. If $\mathcal{A}$ chooses an IND-CCA challenge, he outputs a target identity ID' and two plaintexts M_0, M_1. The challenger tosses a random coin $t \in \{0,1\}$, computes a ciphertext $C' = \mathsf{Encrypt}(cp_{PKG}, ID', M_t)$, and sends C' to $\mathcal{A}$. If $\mathcal{A}$ chooses an EUF-CMA challenge, he continues this game.*
4. *$\mathcal{A}$ continues querying from the decryption, the signature, and the extraction oracles adaptively. If he chooses the IND-CCA challenge, he is neither allowed to query the tuple (ID', C') from the decryption oracle nor ID' from the extract oracle.*
5. *In the end of the game, $\mathcal{A}$ outputs an answer. In the case of the IND-CCA challenge, he outputs a bit t'. $\mathcal{A}$ wins if $t' = t$. In the case of the EUF-CMA challenge, he outputs an identity ID', a message M', and a signature σ'. $\mathcal{A}$ wins if (ID', M') is not asked to the signature oracle, and if $\mathsf{Verify}(cp_{PKG}, ID', M', \sigma') = \mathsf{accept}$.*

Next, we define the key privacy for the ID-based combined scheme. The combined scheme allows users to decrypt ciphertexts and create signatures using the same key. To protect the owner's privacy, the encryption scheme and the signature scheme must satisfy the key privacy condition. When combining them, the encryption scheme should not degrade the key privacy of the signature scheme, and vice versa. Intuitively, the adversary should not be able to distinguish the key even if he is allowed to access the decryption oracle and the signature oracle and even if he receives both a ciphertext and a signature as challenges. We define the key privacy for ID-based scheme when no attacker can distinguish the key in this game.

Definition 13. *The ID-based combined scheme* `IDComb` *is ID-IK secure if no PPT adversary $\mathcal{A}$ has a non-negligible advantage against a challenger in the following game.*

1. *The adversary $\mathcal{A}$ is given the common parameter cp_{PKG}.*
2. *$\mathcal{A}$ is allowed to query to the decryption oracle, the signature oracle, and the extraction oracle adaptively. The decryption oracle receives an identity ID and a ciphertext C and returns the corresponding plaintext $D = \mathsf{Decrypt}(cp_{PKG}, sk_{ID}, C)$. The signature oracle receives an identity ID and a message M and returns the signature $\sigma = \mathsf{Sign}(cp_{PKG}, sk_{ID}, M)$. The extraction oracle receives an identity ID and returns the corresponding secret key S_{ID}.*
3. *When $\mathcal{A}$ announces to the challenger that he is ready, $\mathcal{A}$ outputs two target identities and a plaintext (ID_0, ID_1, M_E). The challenger tosses a random coin $t \in \{0,1\}$ and computes a ciphertext $C' = \mathsf{Encrypt}(cp_{PKG}, ID_t, M_E)$. The challenger also chooses a message $M_S \in \mathcal{M}$ randomly and creates the signature $\sigma' = \mathsf{Sign}(cp_{PKG}, sk_{ID_t}, M_S)$. $\mathcal{A}$ receives C' and σ'.*

4. *$\mathcal{A}$ continues querying from the decryption oracle, the signature oracle, and the extraction oracle adaptively. He is not allowed to query the tuple (ID_0, C') or (ID_1, C') to the decryption oracle, or ID_0 or ID_1 from the extract oracle.*

5. *At the end of the game, $\mathcal{A}$ outputs a bit t'. $\mathcal{A}$ wins if $t' = t$.*

$\mathcal{A}$'s advantage is defined as $\mathbf{Adv} = \left|\Pr[t' = 1|t = 1] - \Pr[t' = 1|t = 0]\right|$. *The probability is taken over the coin tosses of both $\mathcal{A}$ and the challenger, including the coin toss for t.*

4.2 Concrete scheme

We construct a concrete ID-based combined scheme and prove that this scheme satisfies the ID-IND-EUF security and the ID-IK security. This scheme is a combination of the identity-based encryption scheme proposed by Boneh and Franklin [9] and the identity-based signature scheme proposed by Cha and Cheon [15].

Let `IDEnc` be the encryption scheme proposed by Boneh and Franklin, `IDSig` the signature scheme proposed by Cha and Cheon, and `IDComb` our ID-based combined scheme.

In this setting, there are a public key generator PKG and some users. Our ID-based combined scheme `IDComb` consists of the following six algorithms (KeyGen, Extract, Encrypt, Decrypt, Sign, Verify).

- KeyGen: The algorithm chooses two groups $\mathcal{G} = \langle P \rangle$ and $\mathcal{F}$ of the same prime order q. It constructs a bilinear map $\hat{e} : \mathcal{G} \times \mathcal{G} \to \mathcal{F}$. It chooses $s \in \mathbb{Z}_q^*$ uniformly at random and computes $P_{pub} = sP$. Also, it selects hash functions $H_{\mathtt{ID}} : \{0,1\}^* \to \mathcal{G}$, $H_1 : \{0,1\}^* \times \mathcal{G} \to \mathbb{Z}_q$, $H_2 : \mathcal{F} \to \{0,1\}^n$, $H_3 : \{0,1\}^n \times \{0,1\}^n \to \mathbb{Z}_q$, $H_4 : \{0,1\}^n \to \{0,1\}^n$. The system parameter is $(P, P_{pub}, \hat{e}, H_{\mathtt{ID}}, H_1, H_2, H_3, H_4)$. The master key is s. We remark that these hash functions will be viewed as random oracles in our security proof.

- Extract: Given an identity ID, the algorithm computes $S_{ID} = sH_{\mathtt{ID}}(ID)$ and outputs it as the private key associated to ID. We remark that $H_{\mathtt{ID}}(ID)$ plays the role of the associated public key.

- Encrypt: Given the receiver's identity ID and a message $m \in \{0,1\}^n$, the algorithm chooses $\tau \in \{0,1\}^n$ at random, and computes

$$r = H_3(\tau, m), U = rP, V = \tau \oplus H_2(\hat{e}(H_{\mathtt{ID}}(ID), P_{pub})^r), W = m \oplus H_4(\tau).$$

  The ciphertext is $C = (U, V, W)$.

- Decrypt: Given a ciphertext $C = (U, V, W)$ for the identity ID, the algorithm computes

$$\tau = V \oplus H_2(\hat{e}(S_{ID}, U)), m = W \oplus H_4(\tau).$$

  It checks whether $U \stackrel{?}{=} H_3(\tau, m)$. If this equation holds, it returns m as the plaintext. Otherwise, it returns reject.

- Sign: Given a secret key S_{ID} and a message m, the algorithm picks a random number $r \in \mathbb{Z}_q$ and outputs a signature $\sigma = (U, V)$ where $U = rH_{\mathtt{ID}}(ID), V = (r + H_1(m, U))S_{ID}$.

- Verify: To verify a signature $\sigma = (U, V)$ of a message m for an identity ID, the algorithm checks whether $\hat{e}(P_{pub}, U + H_1(m, U)H_{\mathtt{ID}}(ID)) \stackrel{?}{=} \hat{e}(P, V)$. If this equation holds, it returns accept. Otherwise, it returns reject.

Therefore, `IDEnc` = (KeyGen, Extract, Encrypt, Decrypt), `IDSig` = (KeyGen, Extract, Sign, Verify).

4.3 Security analysis

The security of IDEnc in the presence of IDSig: IDComb does not compromise the security of IDEnc.

Proposition 14. *Let H_1 be a random oracle from $\{0,1\}^* \times \mathcal{G}$ to $\mathbb{Z}_q$. Let $\mathcal{A}$ be an adversary that has non-negligible advantage ϵ against* IDEnc *of* IDComb *with unlimited access to H_1 and the signature oracle for* IDSig*. Then, there is an algorithm $\mathcal{B}$ that has non-negligible advantage ϵ' against* IDEnc*.*

The proof is shown in Appendix B.

The security of IDSig in the presence of IDEnc: IDComb does not compromise the security of IDSig.

Proposition 15. *Let H_2 be a random oracle from $\mathcal{F}$ to $\{0,1\}^n$, H_3 a random oracle from $\{0,1\}^n \times \{0,1\}^n$ to $\mathbb{Z}_q$, H_4 a random oracle from $\{0,1\}^n$ to $\{0,1\}^n$. Let $\mathcal{A}$ be an adversary that has non-negligible advantage ϵ against* IDSig *of* IDComb *with unlimited access to H_2, H_3, H_4 and a decryption oracle for* IDEnc*. Then, there is an algorithm $\mathcal{B}$ that has non-negligible advantage ϵ' against* IDSig*.*

The proof is shown in Appendix C.

As a result, we can show the security of IDComb.

Theorem 2. IDComb *is ID-IND-EUF secure by assuming the hardness of the CBDH problem for $\hat{e} : \mathcal{G} \times \mathcal{G} \to \mathcal{F}$ in the random oracle model.*

This proof is shown in Appendix D.

Theorem 3. *The ID-based combined scheme* IDComb *is ID-IK secure by assuming the hardness of the CBDH problem for $\hat{e} : \mathcal{G} \times \mathcal{G} \to \mathcal{F}$ in the random oracle model.*

The proof is shown in Appendix E.

5 The key privacy for combined schemes

In this section, we define the key privacy for the non-ID-based combined schemes. We define the key privacy when no attacker can distinguish the key in this game.

In our combined scheme, two keys are involved in the encryption scheme. Differently from a single-key combined scheme, there are some variations for the requirements for the key privacy. Some people want to protect key information from leaking from both the ciphertexts and the signature. In some cases, protecting one key information from leaking can be required. We classify these security properties to three settings: (i) the key privacy for two keys, (ii) the key privacy for the decryption and signature key, and (iii) the key privacy for the recovery key. We define the key privacy for two keys, and show the alternations for the others. We make a small change for the combined scheme proposed by Diament et al. [16] and prove that this scheme satisfies the key privacy for two keys.

5.1 Model definitions

Definition 16. *A combined scheme is a tuple of six algorithms denoted by $\Sigma = (\mathcal{K}, \mathcal{E}, \mathcal{D}, \mathcal{R}, \mathcal{S}, \mathcal{V})$:*

- *The* key generation algorithm $\mathcal{K}$ *takes a security parameter k as an input, and produces a pair (e, d) of corresponding public and private keys. We write $\mathcal{K}(k) = (e, d)$. Let $\mathcal{K}(k) = (f, g)$ be another key pair in the following.*
- *The* encryption algorithm $\mathcal{E}$ *takes public encryption keys e and f, and a plaintext $m \in \mathcal{M}$ (where $\mathcal{M}$ is the message space) as inputs, and produces a ciphertext $c \in \mathcal{C}$ (where $\mathcal{C}$ is the ciphertext space). We write $\mathcal{E}_{e,f}(m) = c$.*

- *The* decryption algorithm $\mathcal{D}$ *takes a private key* d*, a public key* f*, and a cipher text* $c \in \mathcal{C}$ *as inputs, and produces a plaintext* $m \in \mathcal{M}$ *or a reject symbol* reject*. We write* $\mathcal{D}_{d,f}(c) = m$*.*

- *The* recovery algorithm $\mathcal{R}$ *takes a public key* e*, a private key* g*, and a ciphertext* $c \in \mathcal{C}$ *as inputs, and produces a plaintext* $m \in \mathcal{M}$ *or* reject*. We write* $\mathcal{R}_{e,g}(c) = m$*.*

- *The* signature algorithm $\mathcal{S}$ *takes a private key* d *and a message* $m \in \mathcal{M}$ *as inputs, and produces a signature* $\sigma \in \{0,1\}^*$*. We write* $\mathcal{S}_d(m) = \sigma$*.*

- *The* verification algorithm $\mathcal{V}$ *takes a public key* e*, a message* m*, and a signature* σ *as inputs, and produces either an accept symbol* accept *or* reject*. We write* $\mathcal{V}_e(m, \sigma) =$ accept *or* reject*.*

Next, we define the key privacy for two keys. We consider this key privacy as the key privacy for the combined scheme.

Definition 17. *Let* $\Sigma = (\mathcal{K}, \mathcal{E}, \mathcal{D}, \mathcal{R}, \mathcal{S}, \mathcal{V})$ *be a combined scheme and* k *a security parameter of* Σ*.* Σ *satisfies the key privacy if for all sufficiently large* k*, no PPT adversary* $\mathcal{A}$ *can win the following game with a probability non-negligibly larger than* $1/2$*. This game is simulated by a challenger.*

1. *The challenger runs* $\mathcal{K}(k)$ *four times and receives four pairs of public and private keys. The challenger makes two pairs of public keys* (pk_0, pk'_0) *and* $(pk_1.pk'_1)$ *and gives* $\mathcal{A}$ *them with public parameters.*

2. $\mathcal{A}$ *is allowed to make queries to oracles. The decryption oracle for* (sk_0, pk'_0) *receives a ciphertext* c *and returns a plaintext* $m = \mathcal{D}_{sk_0,pk'_0}(c)$*. The recovery oracle for* (pk_0, sk'_0) *receives a ciphertext* c *and returns a plaintext* $m = \mathcal{R}_{pk_0,sk'_0}(c)$*. The decryption oracle for* (sk_1, pk'_1) *receives a ciphertext* c *and returns a plaintext* $m = \mathcal{D}_{sk_1,pk'_1}(c)$*. The recovery oracle for* (pk_1, sk'_1) *receives a ciphertext* c *and returns a plaintext* $m = \mathcal{R}_{pk_1,sk'_1}(c)$*. The signature oracle for* sk_0 *receives a message* m *and returns a signature* $\sigma = \mathcal{S}_{sk_0}(m)$*. The signature oracle for* sk_1 *receives a message* m *and returns a signature* $\sigma = \mathcal{S}_{sk_1}(m)$*.*

3. *When* $\mathcal{A}$ *is ready for the challenge,* $\mathcal{A}$ *passes a message* m_e*. The challenger tosses a random coin* $t \in \{0,1\}$*, then uniformly picks a message* $m_s \in \mathcal{M}$*. Set* $c' = \mathcal{E}_{pk_t,pk'_t}(m_e), \sigma' = \mathcal{S}_{pk_t}(m_s)$*.* $\mathcal{A}$ *receives* c' *and* σ'*.*

4. $\mathcal{A}$ *can still adaptively make queries to the oracles except for the challenge ciphertext* c' *to the decryption and the recovery oracles.*

5. *At the end of the game,* $\mathcal{A}$ *outputs a bit* t'*.* $\mathcal{A}$ *wins if* $t' = t$*.*

$\mathcal{A}$*'s advantage is defined as* $\mathbf{Adv} = \left|\Pr[t' = t] - \frac{1}{2}\right|$*. The probability is taken over the coin tosses of both* $\mathcal{A}$ *and the challenger, including the coin toss for* t*.*

The modification for the key privacy for the decryption and signature key is as follows: In the step 1 of the game, the challenger runs $\mathcal{K}(k)$ three times. The challenger makes two pairs of public keys (pk_0, pk'_0) and (pk_1, pk'_1) such that $pk'_0 = pk'_1$.

In the same way, the modification for the key privacy for the recovery key is as following: In the step 1 of the game, the challenger runs $\mathcal{K}(k)$ three times. The challenger makes two pairs of public keys (pk_0, pk'_0) and (pk_1, pk'_1) such that $pk_0 = pk_1$.

5.2 Security analysis for our combined scheme

In this section, we introduce a combined scheme Σ modified from the combined scheme proposed by Diament et al. [16]. This modification is to just omit the public key of the recovery person from the ciphertext, which is the term "u_2" in the original scheme. A scheme which has public keys

in the ciphertext cannot be anonymous. We assume that during the legitimate decryption and the recovery operation, they are aware of the receivers. This modification has no impact for the security for the combined scheme which was defined as "CCA-CMA security". Thus Σ satisfies this security, too. The message space $\mathcal{M}$ is $\{0,1\}^n$, the security parameter is n'. b_2 is the length of the bit-representation of a point in $\mathbb{G}_2$.

Key Generation: Groups $\mathbb{G}_1, \mathbb{G}_2$ with a same prime order q, a bilinear map $\langle \cdot, \cdot \rangle : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$, and a random element $P \in \mathbb{G}_1$ are chosen. Choose a random $x \in \mathbb{Z}_q$. The public key is xP together with hash functions $H_x : \mathbb{G}_2 \to \{0,1\}^n$, $G : \{0,1\}^n \to \{0,1\}^n$, $F : \{0,1\}^{4n+b_2} \to \{0,1\}^{n'}$, and $I : \{0,1\}^n \to \mathbb{G}_1$. The private key is x. Let (yP, y) be another pair of public and private keys.

Encryption: The input is a plaintext $m \in \{0,1\}^n$. The encryption algorithm chooses a random element $r \in \mathbb{Z}_q$, a random element $\rho \in \{0,1\}^n$, and computes $u_1 = rP, u_2 = \rho \oplus H_x(\langle xP, yP \rangle^r), u_3 = m \oplus G(\rho)$, and $u_4 = F(\rho, m, u_2, u_3, \langle xP, yP \rangle^r)$. The ciphertext is (u_1, u_2, u_3, u_4).

Decryption: Given a ciphertext (u_1, u_2, u_3, u_4), the decryption algorithm computes $u_2 \oplus H_x(\langle u_1, yP \rangle^x) = \rho$ and $G(\rho) \oplus u_3 = m$. Then it checks that $u_4 = F(\rho, m, u_2, u_3, \langle u_1, yP \rangle^x)$, and if u_4 is correct, the algorithm outputs m. Otherwise, it outputs reject.

Recovery: Given a ciphertext (u_1, u_2, u_3, u_4), the recovery algorithm computes $u_2 \oplus H_x(\langle u_1, xP \rangle^y) = \rho$ and $G(\rho) \oplus u_3 = m$. Then it checks that $u_4 = F(\rho, m, u_2, u_3, \langle u_1, xP \rangle^y)$, and if u_4 is correct, the algorithm outputs m. Otherwise, it outputs reject.

Signature: The input is a private signature key $x \in \mathbb{Z}_q$ and a message $m \in \{0,1\}^n$. The signature algorithm calculates $\sigma = xI(m)$. The signature is σ.

Verification: Given a public key xP and a pair of a message and a signature pair (m, σ), the verification algorithm verifies that $\langle P, \sigma \rangle = \langle xP, I(m) \rangle$.

Proposition 18. *The combined scheme Σ satisfies the key privacy for the combined scheme assuming the CBDH problem is hard in the random oracle model.*

The proof is shown in Appendix F.

6 ID-based combined scheme with multiple-receivers

In this section, we construct a combined scheme of the multi-receiver identity-based encryption scheme proposed by Baek, Safavi-Naini, and Susilo [2] and the identity-based signature scheme proposed by Cha and Cheon [15]. We prove the security of this scheme.

6.1 Model definitions for ID-based combined schemes with multiple-receivers

We adapt the definitions for multi-receivers identity-based encryption schemes originally presented by Baek et al. [2]. In this model, there is a private key generator and some users. Each user has an identity ID and a corresponding private key obtained from the private key generator. Note that in the multi-receiver identity-based encryption setting, either a single message or multiple message can be encrypted. In this paper, we assume that a single message is encrypted.

Definition 19. *An ID-based combined scheme with multiple-receivers consists of six algorithms* MIDComb = (KeyGen, Extract, Encrypt, Decrypt, Sign, Verify) *as follows.*

- *The private key generator's key generation algorithm* KeyGen: *The private key generator runs this algorithm to generate a private key generator's master key and a common parameter, denoted by mk_{PKG}, and cp_{PKG}, respectively. Note that cp_{PKG} is published publicly while mk_{PKG} is kept secret.*

- *The private key generator's private key extraction algorithm* Extract: *Given an identity ID received from a user and the private key generator's master key mk_{PKG} as input, the private key generator runs this algorithm to generate a private key associated with ID, denoted by sk_{ID}. We write $sk_{ID} =$ Extract(mk_{PKG}, ID).*

- *The encryption algorithm* Encrypt: *Given multiple identities* $(ID_1, \ldots, ID_n)$ *of the receivers, the private key generator's common parameter* cp_{PKG}*, and a plaintext* M *as input, the sender runs this algorithm to generate a ciphertext* C *which is an encryption of* M *under* $(ID_1, \ldots, ID_n)$*. We write* $C = \mathsf{Encrypt}(cp_{PKG}, (ID_1, \ldots, ID_n), M)$*.*

- *The decryption algorithm* Decrypt: *Given the receiver's private key* sk_{ID_i}*, the private key generator's common parameter* cp_{PKG}*, and a ciphertext* C *as input, the receiver numbered* i *runs this algorithm to get a decryption* D*, which is either a certain plaintext or a reject symbol* reject*. We write* $D = \mathsf{Decrypt}(cp_{PKG}, sk_{ID_i}, C)$*.*

- *The signature algorithm* Sign: *Given the signer's private key* sk_{ID}*, the private key generator's common parameter* cp_{PKG}*, and a message* M *as input, the signer runs this algorithm to create a signature* σ*. We write* $\sigma = \mathsf{Sign}(cp_{PKG}, sk_{ID}, M)$*.*

- *The verification algorithm* Verify: *Receiving a message* M*, a signature* σ*, and an identity* ID*, the receiver checks the validity of the signature. If it is a valid tuple, it produces an accept symbol* accept*. Otherwise, it produces* reject*. We write* $\mathsf{Verify}(cp_{PKG}, ID, M, \sigma) =$ accept *or* reject*.*

Definition 20. *The ID-based combined scheme with multiple-receivers* `MIDComb` *is* MID-IND-EUF secure *if no PPT adversary* $\mathcal{A}$ *has a non-negligible advantage against a challenger in the following joint MID-IND-EUF game.*

1. *The adversary* $\mathcal{A}$ *is given the common parameter* cp_{PKG}*, then it outputs the target identities* $(ID_1, \ldots, ID_n)$*.*

2. $\mathcal{A}$ *is allowed to query to the decryption, the signature, and the extract oracles adaptively. The* i*-th decryption oracle receives a ciphertext* C *and returns the decryption of* ID_i*,* $D = \mathsf{Decrypt}(cp_{PKG}, sk_{ID_i}, C)$*. The* i*-th signature oracle receives a message* M *and returns the signature* $\sigma = \mathsf{Sign}(cp_{PKG}, sk_{ID_i}, M)$*. The extract oracle receives an identity* ID *and returns the corresponding secret key* S_{ID}*. It is not allowed to query the target identities to the extract oracle.*

3. *When* $\mathcal{A}$ *announces to the challenger that it is ready for the challenge,* $\mathcal{A}$ *chooses either an IND-CCA challenge or an EUF-CMA challenge. If* $\mathcal{A}$ *chooses an IND-CCA challenge, it outputs two plaintexts* M_0, M_1*. The challenger tosses a random coin* $t \in \{0, 1\}$*, compute a ciphertext* $C' = \mathsf{Encrypt}(cp_{PKG}, (ID_1, \ldots, ID_n), M_t)$*, and returns* C *to* $\mathcal{A}$*. If* $\mathcal{A}$ *chooses an EUF-CMA challenge, it continues this game.*

4. $\mathcal{A}$ *continues querying to the decryption, the signature, and the extract oracles adaptively. If it chooses the CCA challenge, it is not allowed to query the challenge* C' *to the decryption oracles.*

5. *In the end of the game,* $\mathcal{A}$ *outputs the answer. In the case of the CCA challenge, a bit* t'*.* $\mathcal{A}$ *wins if* $t' = t$*. In the case of the CMA challenge, an identity* ID_i*, a message* M'*, and a signature* σ'*.* $\mathcal{A}$ *wins if* ID_i *is one of the IDs output in step1, and if* M' *is not asked to the* i*-th signature oracle, and if* $\mathsf{Verify}(cp_{PKG}, ID_i, M', \sigma') =$ accept*.*

6.2 Concrete scheme

We construct a concrete ID-based combined scheme with multiple-receivers. This scheme is a combination of the multi-receiver identity-based encryption scheme proposed by Baek et al. [2] and the identity-based signature scheme proposed by Cha and Cheon [15]. In Baek's setting, either a single message or multiple message can be encrypted. In this paper, we assume that a single message is encrypted to broadcast to the multiple receivers.

Let MIDEnc be the encryption scheme proposed by Baek et al., IDSig the signature scheme proposed by Cha and Cheon, and MIDComb our ID-based combined scheme with multiple-receivers.

In this setting, there are a public key generator PKG and some users. Our ID-based combined scheme MIDComb consists of the following six algorithms (KeyGen, Extract, Encrypt, Decrypt, Sign, Verify).

- KeyGen: Choose two groups $\mathcal{G} = \langle P \rangle$ and $\mathcal{F}$ of the same prime order q. Construct a bilinear map $\hat{e} : \mathcal{G} \times \mathcal{G} \to \mathcal{F}$. Choose $Q \in \mathcal{G}$ uniformly at random. Choose $s \in \mathbb{Z}_q^*$ uniformly at random and compute $P_{pub} = sP$. Also, select a hash function $H_{\mathtt{ID}} : \{0,1\}^* \to \mathcal{G}$, $H_1 : \{0,1\}^* \times \mathcal{G} \to \mathbb{Z}_q$, $H_2 : \mathcal{F} \to \{0,1\}^{k_1}$, $H_3 : \mathcal{F} \times \{0,1\}^{k_1} \times \mathcal{G} \times \cdots \times \mathcal{G} \times \mathcal{F} \times \{0,1\}^{k_1} \to \{0,1\}^{k_2}$. The system parameter is $(P, P_{pub}, \hat{e}, Q, H_{\mathtt{ID}}, H_1, H_2, H_3)$. The master key is s. We remark that these hash functions will be viewed as random oracles in our security proof.

- Extract: Given an identity ID, the algorithm computes $S_{ID} = sH_{\mathtt{ID}}(ID)$ and outputs it as the private key associated to ID. We remark that $H_{\mathtt{ID}}(ID)$ plays the role of the associated public key.

- Encrypt: Given multiple identities $(ID_1, \ldots, ID_n)$ of the receivers and a plaintext $m \in \{0,1\}^{k_1}$, choose $R \in \mathcal{F}$ and $r \in \mathbb{Z}_q$ at random. Let $\mathcal{L}$ be the list of the receivers and computes

$$U = rP, V_1 = rH_{\mathtt{ID}}(ID_1) + rQ, \ldots, V_n = rH_{\mathtt{ID}}(ID_n) + rQ,$$

$$W_1 = \hat{e}(Q, P_{pub})^r R, W_2 = m \oplus H_2(R), \tau = H_3(R, m, U, V_1, \ldots, V_n, W_1, W_2, \mathcal{L}).$$

  The ciphertext is $C = (U, V_1, \ldots, V_n, W_1, W_2, \mathcal{L}, \tau)$.

- Decrypt: Given a ciphertext C for some $ID_i, i \in \{1, \ldots, n\}$, parse C as $(U, V_1, \ldots, V_n, W_1, W_2, \mathcal{L}, \tau)$. Using $\mathcal{L}$, find the appropriate V_i. Then, subsequently compute

$$R = \frac{\hat{e}(U, S_{ID_i})}{\hat{e}(T, V_i)}, m = W_2 \oplus H_2(R).$$

  Check whether $\tau \stackrel{?}{=} H_3(R, M, U, V_1, \ldots, V_n, W_1, W_2, \mathcal{L})$. If this equation holds, return m as a plaintext. Otherwise, return reject.

- Sign: Given a secret key S_{ID} and a message m, pick a random number $r \in \mathbb{Z}_q$ and output a signature $\sigma = (U, V)$ where $U = rH_{\mathtt{ID}}(ID), V = (r + H_1(m, U))S_{ID}$.

- Verify: To verify a signature $\sigma = (U, V)$ of a message m for an identity ID, check whether $\hat{e}(P_{pub}, U + H_1(m, U)H_{\mathtt{ID}}(ID)) \stackrel{?}{=} \hat{e}(P, V)$. If this equation holds, it returns accept. Otherwise, it returns reject.

Therefore, MIDEnc = (KeyGen, Extract, Encrypt, Decrypt), IDSig = (KeyGen, Extract, Sign, Verify).

6.3 Security analysis

The security of MIDEnc in the presence of IDSig: MIDComb does not compromise the security of MIDEnc.

Proposition 21. *Let H_1 be a random oracle from $\{0,1\}^* \times \mathcal{G}$ to $\mathbb{Z}_q$. Let $\mathcal{A}$ be an adversary that has non-negligible advantage ϵ against* MIDEnc *of* MIDComb *with unlimited access to H_1 and a signature oracle for* IDSig*. Then, there is an algorithm $\mathcal{B}$ that has non-negligible advantage ϵ' against* MIDEnc*.*

The proof is shown in Appendix G.

The security of IDSig in the presence of MIDEnc: MIDComb does not compromise the security of IDSig.

Proposition 22. *Let H_2 be a random oracle from $\mathcal{F}$ to $\{0,1\}^{k_1}$, H_3 a random oracle from $\mathcal{F} \times \{0,1\}^{k_1} \times \mathcal{G} \times \cdots \times \mathcal{G} \times \mathcal{F} \times \{0,1\}^{k_1}$ to $\{0,1\}^{k_2}$. Let $\mathcal{A}$ be an adversary that has non-negligible advantage ϵ against* IDSig *of* MIDComb *with unlimited access to H_2, H_3 and a decryption oracle for* IDSig. *Then, there is an algorithm $\mathcal{B}$ that has non-negligible advantage ϵ' against* IDSig.

The proof is shown in Appendix H.

As a result, we can show the security of MIDComb.

Proposition 23. MIDComb *is MID-IND-EUF secure assuming the hardness of the GBDH problem for $\hat{e} : \mathcal{G} \times \mathcal{G} \to \mathcal{F}$ in the random oracle model.*

This proof is shown in Appendix I.

7 Conclusion

We proposed the definition of the key privacy for ID-based signature schemes, the model for ID-based combined scheme. The notion of the key privacy can be modified to the combined schemes, and we showed a concrete scheme. In addition to these discussions, we modeled the key privacy for non-ID-based combined schemes, the security for ID-based combined scheme with multiple-receiver encryption scheme. We reviewed or constructed concrete schemes for each model and proved the security of these schemes.

References

[1] Abdalla, M., Bellare, M., Catalano, D., Kiltz, E., Kohno, T., Lange, T., Malone-Lee, J., Neven, G., Paillier, P., and Shi, H. Searchable encryption revisited: Consistency properties, relation to anonymous ibe, and extensions. In *CRYPTO* (Santa Barbara, California, USA, August 2005), V. Shoup, Ed., vol. 3621 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 205–222.

[2] Baek, J., Safavi-Naini, R., and Susilo, W. Efficient multi-receiver identity-based encryption and its application to broadcast encryption. In *Public Key Cryptography* (Les Diablerets, Switzerland, January 2005), S. Vaudenay, Ed., vol. 3386 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 380–397.

[3] Barreto, P. S. L. M., Libert, B., McCullagh, N., and Quisquater, J.-J. Efficient and provably-secure identity-based signatures and signcryption from bilinear maps. In *ASIACRYPT* (Chennai, India, December 2005), B. Roy, Ed., vol. 3788 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 515–532.

[4] Bellare, M., Boldyreva, A., Desai, A., and Pointcheval, D. Key-privacy in public-key encryption. In *ASIACRYPT* (Gold Coast, Australia, December 2001), C. Boyd, Ed., vol. 2248 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 566–582.

[5] Bellare, M., Namprempre, C., and Neven, G. Security proofs for identity-based identification and signature schemes. In Cachin and Camenisch [13], pp. 268–286.

[6] Boneh, D., and Boyen, X. Efficient selective-id secure identity-based encryption without random oracles. In Cachin and Camenisch [13], pp. 223–238.

[7] Boneh, D., and Boyen, X. Secure identity based encryption without random oracles. In *CRYPTO* (Santa Barbara, California, USA, August 2004), M. Franklin, Ed., vol. 3152 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 443–459.

[8] Boneh, D., Crescenzo, G. D., Ostrovsky, R., and Persiano, G. Public key encryption with keyword search. In Cachin and Camenisch [13], pp. 506–522.

[9] Boneh, D., and Franklin, M. K. Identity-based encryption from the Weil pairing. In *CRYPTO* (Santa Barbara, California, USA, August 2001), J. Kilian, Ed., vol. 2139 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 213–229.

[10] Boneh, D., Gentry, C., Lynn, B., and Shacham, H. Aggregate and verifiably encrypted signatures from bilinear maps. In *EUROCRYPT* (Warsaw, Poland, May 2003), E. Biham, Ed., vol. 2656 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 416–432.

[11] Boyen, X. Multipurpose identity-based signcryption (a swiss army knife for identity-based cryptography). In *CRYPTO* (Santa Barbara, California, USA, August 2003), D. Boneh, Ed., vol. 2729 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 383–399.

[12] Boyen, X., and Waters, B. Anonymous hierarchical identity-based encryption (without random oracles). Cryptology ePrint Archive, Report 2006/085, 2006. http://eprint.iacr.org/.

[13] Cachin, C., and Camenisch, J., Eds. *Advances in Cryptology – EUROCRYPT 2004* (Interlaken, Switzerland, May 2004), vol. 3027 of *Lecture Notes in Computer Science*, Springer-Verlag.

[14] Canetti, R., Halevi, S., and Katz, J. Chosen-ciphertext security from identity-based encryption. In Cachin and Camenisch [13], pp. 207–222.

[15] Cha, J. C., and Cheon, J. H. An identity-based signature from gap Diffie-Hellman groups. In *Public Key Cryptography* (Miami, Florida, USA, January 2003), Y. Desmedt, Ed., vol. 2567 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 18–30.

[16] Diament, T., Lee, H. K., Keromytis, A. D., and Yung, M. The dual receiver cryptosystem and its applications. In *ACM Conference on Computer and Communications Security* (2004), pp. 330–343.

[17] Galindo, D., Herranz, J., and Kiltz, E. On the generic construction of identity-based signatures with additional properties. In *ASIACRYPT* (Shanghai, China, December 2006), L. Xuejia and C. Kefei, Eds., vol. 4284 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 178–193.

[18] Gentry, C. Practical identity-based encryption without random oracles. In *EUROCRYPT* (2006), S. Vaudenay, Ed., vol. 4004 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 445–464.

[19] Gentry, C., and Ramzan, Z. Identity-based aggregate signatures. In Yung et al. [30], pp. 257–273.

[20] Gentry, C., and Silverberg, A. Hierarchical id-based cryptography. In *ASIACRYPT* (Queenstown, New Zealand, December 2002), Y. Zheng, Ed., vol. 2501 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 548–566.

[21] Haber, S., and Pinkas, B. Securely combining public-key cryptosystems. In *ACM Conference on Computer and Communications Security* (2001), pp. 215–224.

[22] Hess, F. Efficient identity based signature schemes based on pairings. In *Selected Areas in Cryptography* (2002), K. Nyberg and H. M. Heys, Eds., vol. 2595 of *Lecture Notes in Computer Science*, Springer, pp. 310–324.

[23] Horwitz, J., and Lynn, B. Toward hierarchical identity-based encryption. In *EUROCRYPT* (Amsterdam, The Netherlands, April 2002), L. Knudsen, Ed., vol. 2332 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 466–481.

[24] Malone-Lee, J., and Mao, W. Two birds one stone: Signcryption using rsa. In *CT-RSA* (San Francisco, CA, USA, April 2003), M. Joye, Ed., vol. 2612 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 211–225.

[25] Paterson, K. G. Dentist 2: 00on codes with low peak-to-average power ratio for multicode cdma. *IEEE Transactions on Information Theory 20*, 3 (2004), 550–559.

[26] Shamir, A. Identity-based cryptosystems and signature schemes. In *CRYPTO* (Santa Barbara, California, USA, August 1984), G. R. Blakley and D. Chaum, Eds., vol. 196 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 47–53.

[27] Smart, N. An identity based authenticated key agreement protocol based on the weil pairing. Cryptology ePrint Archive, Report 2001/111, 2001. http://eprint.iacr.org/.

[28] Waters, B. Efficient identity-based encryption without random oracles. In *EUROCRYPT* (Aarhus, Denmark, May 2005), R. Cramer, Ed., vol. 3494 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 114–127.

[29] Yang, G., Wong, D. S., Deng, X., and Wang, H. Anonymous signature schemes. In Yung et al. [30], pp. 347–363.

[30] Yung, M., Dodis, Y., Kiayias, A., and Malkin, T., Eds. *Public Key Cryptography - PKC 2006, 9th International Conference on Theory and Practice of Public-Key Cryptography* (New York, USA, April 2006), vol. 3958 of *Lecture Notes in Computer Science*, Springer.

A Proof of Theorem 1

Proof. We prove this security by reduction. Let $\mathcal{A}$ be an adversary that breaks the ID-IK-CMA security of `IDSig`, then we construct an adversary $\mathcal{B}$ that solves the discrete logarithm problem in $\mathcal{G}$. Let $\mathcal{A}(D_i) = t$ be an experiment where $\mathcal{A}$ outputs (ID_0, ID_1) and receives a signature $\sigma = \mathsf{Sign}(sk_{ID_j}, M)$ in the challenge phase, then $\mathcal{A}$ outputs t as the answer. Let $\mathbf{Adv}_{\mathcal{A}}$ be $\mathcal{A}$'s advantage. For any identity ID',

$$\begin{aligned}\mathbf{Adv}_{\mathcal{A}} &= \big|\Pr[\mathcal{A}(ID_0) = 1] - \Pr[\mathcal{A}(ID_1) = 1]\big| \\ &\leq \big|\Pr[\mathcal{A}(ID_0) = 1] - \Pr[\mathcal{A}(ID') = 1]\big| + \big|\Pr[\mathcal{A}(ID_1) = 1] - \Pr[\mathcal{A}(ID') = 1]\big|.\end{aligned}$$

Since $\mathcal{A}$'s advantage is non-negligible, $\mathcal{A}$ distinguishes the game $\mathcal{A}(ID_0)$ from $\mathcal{A}(ID')$ or the game $\mathcal{A}(ID_1)$ from $\mathcal{A}(ID')$ with non-negligible probability. Without loss of generality, we can assume that $\mathcal{A}$ distinguishes the game $\mathcal{A}(ID_1)$ from $\mathcal{A}(ID')$.

$\mathcal{B}$ is given (P, aP) as the input of the DL problem. $\mathcal{B}$ sets $P_{pub} = aP$ and sends $(q, \mathcal{G}, \mathcal{F}, \hat{e}, P, P_{pub}, H_{\mathtt{ID}}, H_1)$ to $\mathcal{A}$ where $H_{\mathtt{ID}}$ and H_1 are random oracles controlled by $\mathcal{B}$.

- $H_{\mathtt{ID}}$ queries: $H_{\mathtt{ID}}$ is a random oracle controlled by $\mathcal{B}$. $\mathcal{B}$ keeps a list of tuples, $H_{\mathtt{ID}}$-list. When $\mathcal{A}$ issues a query q_i to $H_{\mathtt{ID}}$, $\mathcal{B}$ checks to see if q_i is on the list. If q_i appears in a tuple (q_i, i_i, r_i), then $\mathcal{B}$ responds with $H_{\mathtt{ID}}(q_i) = i_i$. Otherwise, $\mathcal{B}$ picks a random $r_i \in \mathbb{Z}_q$. $\mathcal{B}$ adds the tuple $(q_i, r_i P, r_i)$ to the $H_{\mathtt{ID}}$-list, and responds with $H_x(q_i) = r_i P$.

- H_1 queries: H_1 is a random oracle controlled by $\mathcal{B}$. $\mathcal{B}$ keeps a list of tuples, H_1-list. When $\mathcal{A}$ issues a query q_i to H_1, $\mathcal{B}$ checks to see if q_i is on the list. If q_i appears in a tuple (q_i, i_i), then $\mathcal{B}$ responds with $H_1(q_i) = i_i$. Otherwise, $\mathcal{B}$ picks a random $r_i \in \mathbb{Z}_q$. $\mathcal{B}$ adds the tuple (q_i, r_i) to the H_1-list, and responds with $H_1(q_i) = r_i$.

- Extract queries: $\mathcal{A}$ queries ID_i to the extract oracle to receive the corresponding secret key S_{ID_i}. $\mathcal{B}$ checks if ID_i is on the $H_{\mathtt{ID}}$-list. If ID_i appears in a tuple (ID_i, i_i, r_i), then $\mathcal{B}$ computes $S_{ID_i} = r_i aP$ and responds S_{ID_i}. Otherwise, $\mathcal{B}$ picks a random $r_i \in \mathbb{Z}_q$. $\mathcal{B}$ adds the tuple $(ID_i, r_i P, r_i)$ to the $H_{\mathtt{ID}}$-list, and responds with $S_{ID} = r_i aP$.

- Signature queries: When $\mathcal{A}$ issues a signature query (ID_i, m_i), $\mathcal{B}$ picks $r_1, r_2 \in \mathbb{Z}_q$ randomly, and sets $U_i = r_1 P - r_2 H_{\mathtt{ID}}(ID), V_i = r_1 P_{pub}$. $\mathcal{B}$ checks if $q_i = (m_i, U_i)$ is on the H_1-list. If q_i appears on the list, $\mathcal{B}$ halts. Otherwise, $\mathcal{B}$ adds the tuple (q_i, r_2) to the H_2-list, and responds $\sigma = (U_i, V_i)$.

When $\mathcal{A}$ announces that it is ready for the challenge, $\mathcal{A}$ outputs two target IDs (ID_0, ID_1). $\mathcal{B}$ chooses $x, y \in \mathbb{Z}_q$ randomly, sets $U' = xH_{\mathtt{ID}}(ID_1), V' = yH_{\mathtt{ID}}(ID_1)$, and sends (U', V') to $\mathcal{A}$ as its challenge.

In the end, $\mathcal{A}$ outputs the answer t'. A signature (U, V) is computed $U = rH_{\mathtt{ID}}(ID), V = (r + H_1(m, U))S_{ID}$. As r is selected from $\mathbb{Z}_p$ randomly, this U is distributed uniformly in $\mathcal{G}$. Since we assumed that H_1 is a random oracle, $V = rS_{ID} + H_1(m, U)S_{ID}$ is distributed uniformly in $\mathcal{G}$. Therefore, any signature (U, V) is independent of the corresponding message and identity without querying (m, U) to H_1. Thus, the probability that $\mathcal{A}$ answers the correct answer without querying the corresponding (m, U') to H_1 is negligible. $\mathcal{B}$ picks a tuple (q_i, r_i) from H_1 such that $q_i = (m, U')$ randomly. This r_i satisfies the following equation.

$$\begin{aligned} V' = yH_{\mathtt{ID}}(ID_1) &= (x + H(m, U'))S_{ID_1} \\ &= (x + r_i)aH_{\mathtt{ID}}(ID_1) \end{aligned}$$

The answer of the DL problem can be computed

$$a = y(x + r_i)^{-1}.$$

Let q_S be the number of the signature queries, and q_{H_1} the number of the H_1 queries. Let Q_{H_1} be the event that $\mathcal{A}$ issues the above query (q_i, r_i) to H_1. $\mathcal{A}$ distinguishes the game $\mathcal{A}(ID_1)$ from $\mathcal{A}(ID')$ with non-negligible probability $\epsilon = \big|\Pr[\mathcal{A}(ID_1) = 1] - \Pr[\mathcal{A}(ID') = 1]\big|$. From the above argument, if $\mathcal{A}$ never issues a correct query for H_1, then the probability that $\mathcal{A}$ answers the correct answer is at most $1/2$. Therefore, $\big|\Pr[\mathcal{A}(ID_1) = 1|\neg Q_{H_1}] - \Pr[\mathcal{A}(ID') = 1|\neg Q_{H_1}]\big| = 0$.

$$\begin{aligned} \epsilon &= \big|\Pr[\mathcal{A}(ID_1) = 1] - \Pr[\mathcal{A}(ID') = 1]\big| \\ &\leq \big|\Pr[\mathcal{A}(ID_1) = 1|Q_{H_1}]\Pr[Q_{H_1}] - \Pr[\mathcal{A}(ID') = 1|Q_{H_1}]\Pr[Q_{H_1}]\big| \\ &\quad + \big|\Pr[\mathcal{A}(ID_1) = 1|\neg Q_{H_1}]\Pr[\neg Q_{H_1}] - \Pr[\mathcal{A}(ID') = 1|\neg Q_{H_1}]\Pr[\neg Q_{H_1}]\big| \\ &= \big|\Pr[\mathcal{A}(ID_1) = 1|Q_{H_1}] - \Pr[\mathcal{A}(ID') = 1|Q_{H_1}]\big|\Pr[Q_{H_1}] \\ &\leq \Pr[Q_{H_1}] \end{aligned}$$

$\mathcal{B}$ halts if the message m and U created in the signature queries are already asked to H_1. The random values r_1, r_2 are chosen uniformly from $\mathbb{Z}_q$, so U is created uniformly random in $\mathcal{G}$. The probability that $\mathcal{B}$ halts during the simulation of the signature oracle is at most $\frac{q_S + q_{H_1}}{|\mathcal{G}|}$. Let ϵ' be the success probability that $\mathcal{B}$ solves DL probability. Then, $\epsilon' \geq \frac{1}{q_{H_1}}\left(1 - \frac{q_S + q_{H_1}}{|\mathcal{G}|}\right)\epsilon$. q_S and q_{H_1} are polynomial but $|\mathcal{G}|$ is not, so ϵ' is non-negligible. Therefore, $\mathcal{B}$ can solve the DL problem with non-negligible probability.

□

B Proof of Proposition 14

Proof. Given an adversary $\mathcal{A}$ that breaks the indistinguishability of `IDEnc` when used together with `IDSig` with non-negligible probability ϵ, we construct an adversary $\mathcal{B}$ that breaks the indistinguishability of `IDEnc` alone.

Algorithm $\mathcal{B}$ is given $(q, \mathcal{G}, \mathcal{F}, \hat{e}, P, P_{pub}, H_{\mathtt{ID}}, H_2, H_3, H_4)$, and $\mathcal{B}$ sends $(q, \mathcal{G}, \mathcal{F}, \hat{e}, P, P_{pub}, H_{\mathtt{ID}}, H_1, H_2, H_3, H_4$ to $\mathcal{A}$ where H_1 is a random oracle controlled by $\mathcal{B}$.

- H_1 queries: H_1 is a random oracle controlled by $\mathcal{B}$. $\mathcal{B}$ keeps a list of tuples, H_1-list. When $\mathcal{A}$ issues a query q_i to H_1, $\mathcal{B}$ checks to see if q_i is on the list. If q_i appears in a tuple (q_i, i_i), then $\mathcal{B}$ responds with $H_1(q_i) = i_i$. Otherwise, $\mathcal{B}$ picks a random $r_i \in \mathbb{Z}_q$. $\mathcal{B}$ adds the tuple (q_i, r_i) to the H_1-list, and responds with $H_1(q_i) = r_i$.
- Signature queries: When $\mathcal{A}$ issues a signature query (ID_i, m_i), $\mathcal{B}$ picks $r_1, r_2 \in \mathbb{Z}_q$ randomly, and sets $U_i = r_1 P - r_2 H_{\mathtt{ID}}(ID), V_i = r_1 P_{pub}$. $\mathcal{B}$ checks if $q_i = (m_i, U_i)$ is on the H_1-list. If q_i appears on the list, $\mathcal{B}$ halts. Otherwise, $\mathcal{B}$ adds the tuple (q_i, r_2) to the H_2-list, and responds $\sigma = (U_i, V_i)$.

$\mathcal{A}$'s view of the ciphertext is identical to that of a real ciphertext. Thus, if $\mathcal{B}$ does not halt, its probability of success in breaking the signature scheme is unchanged. Let q_S be the number of the signature queries, and q_{H_1} the number of the H_1 queries. $\mathcal{B}$ halts if the message m and U created in the signature queries is asked to H_1. The random values r_1, r_2 are chosen uniformly from $\mathbb{Z}_q$, so U is created uniformly random in $\mathcal{G}$. The probability that $\mathcal{B}$ halts during the simulation of the signature oracle is at most $\frac{q_S + q_{H_1}}{|\mathcal{G}|}$. The success probability that $\mathcal{B}$ breaks the indistinguishability of `IDEnc` is $\epsilon' \geq \epsilon \left(1 - \frac{q_S + q_{H_1}}{|\mathcal{G}|}\right)$. q_S and q_{H_1} are polynomial but $|\mathcal{G}|$ is not, so ϵ' is non-negligible. □

C Proof of Proposition 15

Proof. Given an adversary $\mathcal{A}$ that breaks the unforgeability of `IDSig` when used together with `IDEnc`, we construct an adversary $\mathcal{B}$ that breaks the unforgeability of `IDSig` alone.

Algorithm $\mathcal{B}$ is given $(q, \mathcal{G}, \mathcal{F}, \hat{e}, P, P_{pub}, H_{\mathtt{ID}}, H_1)$, and $\mathcal{B}$ sends $(q, \mathcal{G}, \mathcal{F}, \hat{e}, P, P_{pub}, H_{\mathtt{ID}}, H_1, H_2, H_3, H_4)$ to $\mathcal{A}$ where H_2, H_3, and H_4 are random oracles controlled by $\mathcal{B}$.

- H_2 queries: H_2 is a random oracle controlled by $\mathcal{B}$. $\mathcal{B}$ keeps a list of tuples, H_2-list. When $\mathcal{A}$ issues a query q_i to H_2, $\mathcal{B}$ checks to see if q_i is on the list. If q_i appears in a tuple (q_i, i_i), then $\mathcal{B}$ responds with $H_2(q_i) = i_i$. Otherwise, $\mathcal{B}$ picks a random $r_i \in \{0,1\}^n$. $\mathcal{B}$ adds the tuple (q_i, r_i) to the H_2-list, and responds with $H_2(q_i) = r_i$.
- H_3 queries: H_3 is a random oracle controlled by $\mathcal{B}$, too. $\mathcal{B}$ keeps a list of tuples, H_3-list. When $\mathcal{A}$ issues a query q_i to H_3, $\mathcal{B}$ checks to see if q_i is on the list. If q_i appears in a tuple (q_i, i_i), then $\mathcal{B}$ responds with $H_3(q_i) = i_i$. Otherwise, $\mathcal{B}$ picks a random $r_i \in \mathbb{Z}_q$. $\mathcal{B}$ adds the tuple (q_i, r_i) to the H_3-list, and responds with $H_3(q_i) = r_i$.
- H_4 queries: H_4 is a random oracle controlled by $\mathcal{B}$, too. $\mathcal{B}$ keeps a list of tuples, H_4-list. When $\mathcal{A}$ issues a query q_i to H_4, $\mathcal{B}$ checks to see if q_i is on the list. If q_i appears in a tuple (q_i, i_i), then $\mathcal{B}$ responds with $H_4(q_i) = i_i$. Otherwise, $\mathcal{B}$ picks a random $r_i \in \{0,1\}^n$. $\mathcal{B}$ adds the tuple (q_i, r_i) to the H_4-list, and responds with $H_4(q_i) = r_i$.
- Decryption queries: $\mathcal{A}$ allows to access the decryption oracles using S_{ID}. $\mathcal{B}$ responds as follows when $\mathcal{A}$ queries $C_i = (U, V, W)$ to the decryption oracle using S_{ID}, $\mathcal{B}$ checks if the tuple (τ, m) that satisfies $H_3(\tau, m)P = U$ is on the H_3-list. If it is not on the list, $\mathcal{B}$ responds reject. Otherwise, sets $h_2 = \tau \oplus V$, $h_4 = m \oplus W$, checks if (q_i, h_2) is on the H_2-list, and

if (τ, h_4) is on the H_4-list. If the both pairs appear on the lists, $\mathcal{B}$ responds m. If τ is not queried to the H_4, $\mathcal{B}$ adds the tuple (τ, h_4) to the H_4-list, and responds m. Otherwise, $\mathcal{B}$ responds reject.

$\mathcal{A}$'s view of the signature is identical to that of a real signature, and thus its probability of success in breaking the encryption scheme is unchanged, $\epsilon' = \epsilon$. □

D Proof of Theorem 2

Proof. This proof is trait-forward. We prove this security by the reduction.

Let $\mathcal{A}$ be an adversary that breaks the ID-IND-EUF security of IDComb. $\mathcal{A}$ breaks the indistinguishability or the existential unforgeability of IDComb. From Lemma 14 and Lemma 15, we can construct an attacker $\mathcal{A}'$ that breaks the indistinguishability of IDEnc alone or $\mathcal{A}''$ that breaks the existential unforgeability of IDSig alone with non-negligible probability. The security of IDEnc is proved by Boneh and Franklin [9], and by using $\mathcal{A}'$, the CBDH problem can be solved. The security of IDSig is proved by Cha and Cheon [15]. By using $\mathcal{A}''$, the CDH problem in $\mathcal{G}$ can be solved. Which means that on input of (P, aP, bP), abP can be computed. The answer of the CBDH problem is computed by $\hat{e}(P,P)^{abc} = \hat{e}(abP, cP)$. □

E Proof of Theorem 3

Proof. Let $\mathcal{A}$ be an adversary that breaks the ID-IK security of IDComb. Let $\mathcal{A}(i,j) = t$ be an experiment where $\mathcal{A}$ outputs (ID_0, ID_1, m) and receives an ciphertext $C' = \mathsf{Encrypt}(ID_i, m)$ and a signature $\sigma = \mathsf{Sign}(sk_{ID_j}, M_S)$ in the challenge phase, then $\mathcal{A}$ outputs t as the answer. Let $\mathbf{Adv}_{\mathcal{A}}$ be $\mathcal{A}$'s advantage. Then,

$$\begin{aligned} \mathbf{Adv}_{\mathcal{A}} &= \big|\Pr[\mathcal{A}(0,0)=1] - \Pr[\mathcal{A}(1,1)=1]\big| \\ &\leq \big|\Pr[\mathcal{A}(0,0)=1] - \Pr[\mathcal{A}(0,1)=1]\big| + \big|\Pr[\mathcal{A}(0,1)=1] - \Pr[\mathcal{A}(1,1)=1]\big|. \end{aligned} \tag{1}$$

Before proving Theorem 3, we introduce the following two propositions.

Proposition 24. *Assume that $\mathcal{A}$ distinguishes the key between the experiments $\mathcal{A}(0,0)$ and $\mathcal{A}(0,1)$ with non-negligible probability, then there is an adversary $\mathcal{B}$ which breaks the key privacy of* IDSig *with non-negligible probability.*

Proof. (of Proposition 24.) Given an adversary $\mathcal{A}$ distinguishes the key between the experiments $\mathcal{A}(0,0)$ and $\mathcal{A}(0,1)$, we construct an adversary $\mathcal{B}$ which breaks the key privacy of IDSig with non-negligible probability.

Algorithm $\mathcal{B}$ is given $(q, \mathcal{G}, \mathcal{F}, \hat{e}, P, P_{pub}, H_{\mathtt{ID}}, H_1)$, and $\mathcal{B}$ sends $(q, \mathcal{G}, \mathcal{F}, \hat{e}, P, P_{pub}, H_{\mathtt{ID}}, H_1, H_2, H_3, H_4)$ to $\mathcal{A}$ where H_2, H_3, and H_4 are random oracles controlled by $\mathcal{B}$.

- H_2 queries: H_2 is a random oracle controlled by $\mathcal{B}$. $\mathcal{B}$ keeps a list of tuples, H_2-list. When $\mathcal{A}$ issues a query q_i to H_2, $\mathcal{B}$ checks to see if q_i is on the list. If q_i appears in a tuple (q_i, i_i), then $\mathcal{B}$ responds with $H_2(q_i) = i_i$. Otherwise, $\mathcal{B}$ picks a random $r_i \in \{0,1\}^n$. $\mathcal{B}$ adds the tuple (q_i, r_i) to the H_2-list, and responds with $H_2(q_i) = r_i$.
- H_3 queries: H_3 is a random oracle controlled by $\mathcal{B}$, too. $\mathcal{B}$ keeps a list of tuples, H_3-list. When $\mathcal{A}$ issues a query q_i to H_3, $\mathcal{B}$ checks to see if q_i is on the list. If q_i appears in a tuple (q_i, i_i), then $\mathcal{B}$ responds with $H_3(q_i) = i_i$. Otherwise, $\mathcal{B}$ picks a random $r_i \in \mathbb{Z}_q$. $\mathcal{B}$ adds the tuple (q_i, r_i) to the H_3-list, and responds with $H_3(q_i) = r_i$.

- H_4 queries: H_4 is a random oracle controlled by $\mathcal{B}$, too. $\mathcal{B}$ keeps a list of tuples, H_4-list. When $\mathcal{A}$ issues a query q_i to H_4, $\mathcal{B}$ checks to see if q_i is on the list. If q_i appears in a tuple (q_i, i_i), then $\mathcal{B}$ responds with $H_4(q_i) = i_i$. Otherwise, $\mathcal{B}$ picks a random $r_i \in \{0,1\}^n$. $\mathcal{B}$ adds the tuple (q_i, r_i) to the H_4-list, and responds with $H_4(q_i) = r_i$.

- Decryption queries: $\mathcal{A}$ allows to access the decryption oracles using S_{ID}. $\mathcal{B}$ responds as follows when $\mathcal{A}$ queries $C_i = (U, V, W)$ to the decryption oracle using S_{ID}, $\mathcal{B}$ checks if the tuple (τ, m) that satisfies $H_3(\tau, m)P = U$ is on the H_3-list. If it is not on the list, $\mathcal{B}$ responds reject. Otherwise, sets $h_2 = \tau \oplus V$, $h_4 = m \oplus W$, checks if (q_i, h_2) is on the H_2-list, and if (τ, h_4) is on the H_4-list. If the both pairs appear on the lists, $\mathcal{B}$ responds m. If τ is not queried to the H_4, $\mathcal{B}$ adds the tuple (τ, h_4) to the H_4-list, and responds m. Otherwise, $\mathcal{B}$ responds reject.

When $\mathcal{A}$ announces that it is ready for the challenge, $\mathcal{A}$ outputs two target IDs and a plaintext (ID_0, ID_1, m). $\mathcal{B}$ then outputs (ID_0, ID_1) to its challenge oracle and receives a signature σ. $\mathcal{B}$ computes $C = \mathsf{Encrypt}(ID_0, m)$, and sends (C, σ) to $\mathcal{A}$ as its challenge.

$\mathcal{A}$'s view of the signature is identical to that of a real signature, and thus its probability of success to distinguish the key is unchanged. Therefore, $\mathbf{Adv}_{\mathcal{B}} = \mathbf{Adv}_{\mathcal{A}}$. □

Proposition 25. *Assume that $\mathcal{A}$ distinguishes the key between the experiments $\mathcal{A}(0,1)$ and $\mathcal{A}(1,1)$ with non-negligible probability, then there is an adversary $\mathcal{C}$ which breaks the key privacy of* IDEnc *with non-negligible probability.*

Proof. (of Proposition 25.) Given an adversary $\mathcal{A}$ distinguishes the key between the experiments $\mathcal{A}(0,1)$ and $\mathcal{A}(1,1)$, we construct an adversary $\mathcal{C}$ which breaks the key privacy of IDEnc with non-negligible probability.

Algorithm $\mathcal{C}$ is given $(q, \mathcal{G}, \mathcal{F}, \hat{e}, P, P_{pub}, H_{\mathtt{ID}}, H_2, H_3, H_4)$, and $\mathcal{C}$ sends $(q, \mathcal{G}, \mathcal{F}, \hat{e}, P, P_{pub}, H_{\mathtt{ID}}, H_1, H_2, H_3, H_4)$ to $\mathcal{A}$ where H_1 is a random oracle controlled by $\mathcal{C}$.

- H_1 queries: H_1 is a random oracle controlled by $\mathcal{C}$. $\mathcal{C}$ keeps a list of tuples, H_1-list. When $\mathcal{A}$ issues a query q_i to H_1, $\mathcal{C}$ checks to see if q_i is on the list. If q_i appears in a tuple (q_i, i_i), then $\mathcal{C}$ responds with $H_1(q_i) = i_i$. Otherwise, $\mathcal{C}$ picks a random $r_i \in \mathbb{Z}_q$. $\mathcal{C}$ adds the tuple (q_i, r_i) to the H_1-list, and responds with $H_1(q_i) = r_i$.

- Signature queries: When $\mathcal{A}$ issues a signature query (ID_i, m_i), $\mathcal{C}$ picks $r_1, r_2 \in \mathbb{Z}_q$ randomly, and sets $U_i = r_1 P - r_2 H_{\mathtt{ID}}(ID), V_i = r_1 P_{pub}$. $\mathcal{C}$ checks if $q_i = (m_i, U_i)$ is on the H_1-list. If q_i appears on the list, $\mathcal{C}$ halts. Otherwise, $\mathcal{C}$ adds the tuple (q_i, r_2) to the H_2-list, and responds $\sigma = (U_i, V_i)$.

When $\mathcal{A}$ announces that it is ready for the challenge, $\mathcal{A}$ outputs two target IDs and a plaintext (ID_0, ID_1, m). $\mathcal{C}$ then outputs (ID_0, ID_1, m) to its challenge oracle and receives a ciphertext C. $\mathcal{C}$ chooses a message $m' \in \{0,1\}^n$ randomly, computes $\sigma = \mathsf{Sign}(ID_1, m')$, and sends (C, σ) to $\mathcal{A}$ as its challenge.

$\mathcal{A}$'s view of the ciphertext is identical to that of a real ciphertext. Thus, if $\mathcal{C}$ does not halt, its probability of success to distinguish the key is unchanged. Let q_S be the number of the signature queries, and q_{H_1} the number of the H_1 queries. $\mathcal{C}$ halts if the message m and U created in the signature queries are asked to H_1. The random values r_1, r_2 are chosen uniformly from $\mathbb{Z}_q$, so U is created uniformly random in $\mathcal{G}$. The probability that $\mathcal{C}$ halts during the simulation of the signature oracle is at most $\frac{q_S + q_{H_1}}{|\mathcal{G}|}$. The success probability that $\mathcal{C}$ breaks the indistinguishability of IDEnc is $\mathbf{Adv}_{\mathcal{C}} \geq \mathbf{Adv}_{\mathcal{A}} \left(1 - \frac{q_S + q_{H_1}}{|\mathcal{G}|}\right)$. q_S and q_{H_1} are polynomial but $|\mathcal{G}|$ is not, so $\mathbf{Adv}_{\mathcal{C}}$ is non-negligible.

□

The rest of the proof of Theorem 3 is as follows. Assume that there exist an attacker $\mathcal{A}$ which breaks the key privacy of `IDComb` with non-negligible probability. Using the above equation 1, there exists an adversary $\mathcal{B}$ which breaks the key privacy of `IDEnc` or an adversary $\mathcal{C}$ which breaks the key privacy of `IDSig`. The key privacy of `IDEnc` is proved by Abdalla et al. [1]. Using this proof, the CBDH problem can be solved with non-negligible probability. The key privacy of `IDSig` is proved in Section 3. This proof showed the construction of an adversary to solve the DL problem in $\mathcal{G}$. This means that on the input of (P, aP), $a \in \mathbb{Z}_q$ can be computed. Using this adversary, the CBDH problem can be solved by $\hat{e}(P,P)^{abc} = \hat{e}(bP, cP)^a$.

Therefore, we can solve the CBDH problem with non-negligible probability.

□

F Proof of Proposition 18

Proof. Given an adversary $\mathcal{A}$ that attacks the key privacy of Σ, we construct an adversary $\mathcal{B}$ that solves CBDH problem.

Algorithm $\mathcal{B}$ is given (P, aP, bP, cP), and chooses random elements $x, y \in \mathbb{Z}_q$. $\mathcal{B}$ chooses a random $t \in \{0,1\}$, and sets $\tilde{t} = 1 - t, pk_t = aP, pk'_t = bP, pk_{\tilde{t}} = xP, pk'_{\tilde{t}} = yP$. $\mathcal{B}$ sends $(P, (pk_0, pk'_0), (pk_1, pk'_1), H_x, G, F, I)$ to $\mathcal{A}$ where H_x, G, F, I are random oracles controlled by $\mathcal{B}$.

- H_x queries: H_x is a random oracle controlled by $\mathcal{B}$. $\mathcal{B}$ keeps a list of tuples, H_x-list. When $\mathcal{A}$ issues a query q_i to H_x, $\mathcal{B}$ checks to see if q_i is on the list. If q_i appears in a tuple (q_i, i_i), then $\mathcal{B}$ responds with $H_x(q_i) = i_i$. Otherwise, $\mathcal{B}$ picks a random $r_i \in \{0,1\}^n$. $\mathcal{B}$ adds the tuple (q_i, r_i) to the H_x-list, and responds with $H_x(q_i) = r_i$.

- G queries: G is a random oracle controlled by $\mathcal{B}$, too. $\mathcal{B}$ keeps a list of tuples, G-list. When $\mathcal{A}$ issues a query q_i to G, $\mathcal{B}$ checks to see if q_i is on the list. If q_i appears in a tuple (q_i, i_i), then $\mathcal{B}$ responds with $G(q_i) = i_i$. Otherwise, $\mathcal{B}$ picks a random $r_i \in \{0,1\}^n$. $\mathcal{B}$ adds the tuple (q_i, r_i) to the G-list, and responds with $G(q_i) = r_i$.

- F queries: F is a random oracle controlled by $\mathcal{B}$, too. $\mathcal{B}$ keeps a list of tuples, F-list. When $\mathcal{A}$ issues a query q_i to F, $\mathcal{B}$ checks to see if q_i is on the list. If q_i appears in a tuple (q_i, i_i), then $\mathcal{B}$ responds with $F(q_i) = i_i$. Otherwise, $\mathcal{B}$ picks a random $r_i \in \{0,1\}^{n'}$. $\mathcal{B}$ adds the tuple (q_i, r_i) to the F-list, and responds with $F(q_i) = r_i$.

- I queries: I is a random oracle controlled by $\mathcal{B}$, too. $\mathcal{B}$ keeps a list of tuples, I-list. When $\mathcal{A}$ issues a query q_i to I, $\mathcal{B}$ checks to see if q_i is on the list. If q_i appears in a tuple (q_i, i_i, r_i), then $\mathcal{B}$ responds with $I(q_i) = i_i$. Otherwise, $\mathcal{B}$ picks a random $r_i \in \mathbb{Z}_q$. $\mathcal{B}$ adds the tuple $(q_i, r_i P, r_i)$ to the I-list, and responds with $I(q_i) = r_i P$.

- Decryption and recovery queries: $\mathcal{A}$ allows to access the decryption oracles using (sk_0, pk'_0) and (sk_1, pk'_1), and the recovery oracles using (pk_0, sk'_0) and (pk_1, sk'_1). $\mathcal{B}$ responds as follows when $\mathcal{A}$ issues the decryption query $C_i = (u_1, u_2, u_3, u_4)$. If $\mathcal{A}$ queries to the recovery oracles, $\mathcal{B}$ responds the same way to the each decryption oracle. If $\mathcal{A}$ queries to the decryption oracle using $(sk_{\tilde{t}}, pk'_{\tilde{t}})$, $\mathcal{B}$ decrypts C_i with the secret key x, and responds the plaintext. If $\mathcal{A}$ queries to the decryption oracle using (sk_t, pk'_t), $\mathcal{B}$ checks if the tuple (q_i, u_4) is on the F-list. If it is on the list, sets $g = R$ which is the last term of q_i. Otherwise, picks a random $g \in \mathbb{G}_2$. $\mathcal{B}$ then checks if (g, h_i) is on the H_x-list. If h_i appears on the list, $\mathcal{B}$ computes $\rho = u_2 \oplus h_i$. Otherwise, $\mathcal{B}$ picks $h_i \in \{0,1\}^n$ randomly, add (g, h_i) to the H_x-list, and computes $\rho = u_2 \oplus h_i$. $\mathcal{B}$ checks if (ρ, j) is on the G-list. If j appears on the list, $\mathcal{B}$ computes $m = u_3 \oplus j$. Otherwise, $\mathcal{B}$ picks $j \in \{0,1\}^n$ randomly, add (ρ, j) to the G-list, and computes $m = u_3 \oplus j$. If the (q_i, u_4) was not on the H_x-list, add $((\rho, m, u_2, u_3, g), u_4)$ to the H_x-list. $\mathcal{B}$ responds m as the decryption of C_i.

- Signature queries: When $\mathcal{A}$ issues a signature query to the signature oracle using $sk_{\tilde{i}}$, $\mathcal{B}$ creates the signature with the secret key x, and return the signature. When $\mathcal{A}$ issues a signature query m to the signature oracle using sk_t, $\mathcal{B}$ checks if (m, i_i, r_i) is on the I-list. If it appears on the list, $\mathcal{B}$ computes $\sigma = r_i aP$. Otherwise, picks a random $r_i \in \mathbb{Z}_q$, adds the tuple $(m, r_i P, r_i)$ to the I-list, and computes $\sigma = r_i aP$. $\mathcal{B}$ responds σ.

When $\mathcal{A}$ is ready for the challenge, $\mathcal{A}$ outputs a target plaintext m to the challenge oracle. $\mathcal{B}$ then chooses a random $r' \in \mathbb{Z}_q$ and computes the signature $\sigma' = r'P$. $\mathcal{B}$ sets $u_1 = cP$ and choose randomly $u_2, u_3 \in \{0,1\}^n$, $u_4 \in \{0,1\}^{n'}$. The challenge ciphertext C is (u_1, u_2, u_3, u_4). $\mathcal{B}$ returns (C, σ') as a challenge.

In the end, $\mathcal{A}$ outputs an answer t'. Since $\mathcal{A}$ can win the game with non-negligible probability, it creates the queries associated to the challenges. With probability $1/2$, assume that it revealed the key information from the ciphertext. In this case, $\mathcal{A}$ queries $\langle u_1, bP\rangle^a = \langle cP, bP\rangle^a = \langle P, P\rangle^{abc}$ or $\langle u_1, aP\rangle^b = \langle cP, aP\rangle^b = \langle P, P\rangle^{abc}$ to the H_x-list. $\mathcal{B}$ picks (q_i, h_i) from the H_x-list randomly, and outputs q_i as the answer of the CBDH problem. Otherwise, assume that it revealed the key information form the signature. In this case, $\mathcal{A}$ queries m' such that the equation $\langle P, \sigma'\rangle = \langle P_{pub}, I(m')\rangle$ holds. Let $I(m') = r''$. From this equation, $\langle P, \sigma'\rangle = \langle aP, r''P\rangle = \langle P, P\rangle^{ar''}$. Thus $a = r' \cdot r''^{-1}$ and $\langle P, P\rangle^{abc} = \langle bP, cP\rangle^{r' \cdot r''^{-1}}$. $\mathcal{B}$ picks (q_i, i_i, r_i) from the I-list randomly, and outputs $\langle bP, cP\rangle^{r' \cdot r''^{-1}}$ as the answer of the CBDH problem.

Let Q_{H_x} be the event that $\mathcal{A}$ issues a query including $\langle P, P\rangle^{abc}$ to H_x, Q_I the event that $\mathcal{A}$ issues a query $I(m') = r''P$ to I. Let q_h the number of the queries to H_x, q_I the number of the queries to I. $\mathcal{A}$ wins this game with non-negligible advantage $\epsilon = \Pr[t' = t] - \frac{1}{2}$. In the random oracle mode, if $\mathcal{A}$ never issues a correct query for H_x and I, then the decryption of the ciphertext and the message for the signature is independent from the ciphertext and the signature. Therefore, $\Pr[t' = t | \neg Q_{H_x} \wedge \neg Q_I] = \frac{1}{2}$.

$$\begin{aligned}\Pr[b' = b] &\leq \Pr[t' = t | Q_{H_x}] \Pr[Q_{H_x}] + \Pr[t' = t | Q_I] \Pr[Q_I] + \Pr[t' = t | \neg Q_{H_x} \wedge \neg Q_I] \\ &\leq \Pr[Q_{H_x}] + \Pr[Q_I] + \frac{1}{2}\end{aligned}$$

The probability ϵ' that $\mathcal{B}$ produces the correct answer is $\frac{1}{2}\left(\frac{\Pr[Q_{H_x}]}{q_H} + \frac{\Pr[Q_I]}{q_I}\right)$. Then,

$$\begin{aligned}\epsilon' &= \frac{1}{2}\left(\frac{\Pr[Q_{H_x}]}{q_H} + \frac{\Pr[Q_I]}{q_I}\right) \\ &\geq \frac{1}{2(q_H + q_I)}(\Pr[Q_{H_x}] + \Pr[Q_I]) \\ &\geq \frac{1}{2(q_H + q_I)}\epsilon.\end{aligned}$$

Thus, $\mathcal{B}$ answers the CBDH problem with non-negligible probability. □

G Proof of Proposition 21

Proof. Given an adversary $\mathcal{A}$ that attacks `MIDEnc` when used together with `IDSig`, we construct an adversary $\mathcal{B}$ attacking `MIDEnc` alone.

Algorithm $\mathcal{B}$ is given $(q, \mathcal{G}, \mathcal{F}, \hat{e}, P, Q, P_{pub}, H_{\mathtt{ID}}, H_2, H_3)$, and $\mathcal{B}$ sends $(q, \mathcal{G}, \mathcal{F}, \hat{e}, P, Q, P_{pub}, H_{\mathtt{ID}}, H_1, H_2, H_3)$ to $\mathcal{A}$ where H_1 is a random oracle controlled by $\mathcal{B}$.

- H_1 queries: H_1 is a random oracle controlled by $\mathcal{B}$. $\mathcal{B}$ keeps a list of tuples, H_1-list. When $\mathcal{A}$ issues a query q_i to H_1, $\mathcal{B}$ checks to see if q_i is on the list. If q_i appears in a tuple (q_i, i_i), then $\mathcal{B}$ responds with $H_1(q_i) = i_i$. Otherwise, $\mathcal{B}$ picks a random $r_i \in \mathbb{Z}_q$. $\mathcal{B}$ adds the tuple (q_i, r_i) to the H_1-list, and responds with $H_1(q_i) = r_i$.

- Signature queries: When $\mathcal{A}$ issues a signature query (ID_i, m), $\mathcal{B}$ picks random $r_1, r_2 \in \mathbb{Z}_q$, and sets $U_i = r_1 P - r_2 H_{\mathtt{ID}}(ID_i), V_i = r_1 P$. $\mathcal{B}$ checks if $q_j = (m_i, U_i)$ is on the H_1-list. If q_j appears on the list, $\mathcal{B}$ halts. Otherwise, $\mathcal{B}$ adds the tuple (q_j, r_2) to the H_2-list, and responds $\sigma = (U_i, V_i)$.

$\mathcal{A}$'s view of the ciphertext is identical to that of a real ciphertext, and thus if $\mathcal{B}$ does not halt, its probability of success in breaking the signature scheme is unchanged. Let q_S be the number of the signature queries, and q_{H_1} the number of the H_1 queries. $\mathcal{B}$ halts if the message m and U created in the decryption queries is asked to H_1. The random values r_1, r_2 are chosen uniformly from $\mathbb{Z}_q$, so U is created uniformly random in $\mathcal{G}$. The probability that $\mathcal{B}$ halts during the simulation of the signature oracle is at most $\frac{q_S + q_{H_1}}{|\mathcal{G}|}$. The success probability of $\mathcal{B}$ is $\epsilon' \geq \epsilon \left(1 - \frac{q_S + q_{H_1}}{|\mathcal{G}|}\right)$. q_S and q_{H_1} are polynomial but $|\mathcal{G}|$ is not, so ϵ' is non-negligible. □

H Proof of Proposition 22

Proof. Given an adversary $\mathcal{A}$ that attacks `IDSig` when used together with `MIDEnc`, we construct an adversary $\mathcal{B}$ attacking `IDSig` alone.

Algorithm $\mathcal{B}$ is given $(q, \mathcal{G}, \mathcal{F}, \hat{e}, P, P_{pub}, H_{\mathtt{ID}}, H_1)$, and choose a random element $Q \in \mathcal{G}$. $\mathcal{B}$ sends $(q, \mathcal{G}, \mathcal{F}, \hat{e}, P, Q, P_{pub}, H_{\mathtt{ID}}, H_1, H_2, H_3)$ to $\mathcal{A}$ where H_2, H_3 are random oracles controlled by $\mathcal{B}$.

- H_2 queries: H_2 is a random oracle controlled by $\mathcal{B}$. $\mathcal{B}$ keeps a list of tuples, H_2-list. When $\mathcal{A}$ issues a query q_i to H_2, $\mathcal{B}$ checks to see if q_i is on the list. If q_i appears in a tuple (q_i, i_i), then $\mathcal{B}$ responds with $H_2(q_i) = i_i$. Otherwise, $\mathcal{B}$ picks a random $r_i \in \{0,1\}^{k_1}$. $\mathcal{B}$ adds the tuple (q_i, r_i) to the H_2-list, and responds with $H_2(q_i) = r_i$.

- H_3 queries: H_3 is a random oracle controlled by $\mathcal{B}$, too. $\mathcal{B}$ keeps a list of tuples, H_3-list. When $\mathcal{A}$ issues a query q_i to H_3, $\mathcal{B}$ checks to see if q_i is on the list. If q_i appears in a tuple (q_i, i_i), then $\mathcal{B}$ responds with $H_3(q_i) = i_i$. Otherwise, $\mathcal{B}$ picks a random $r_i \in \{0,1\}^{k_2}$. $\mathcal{B}$ adds the tuple (q_i, r_i) to the H_3-list, and responds with $H_3(q_i) = r_i$.

- Decryption queries: $\mathcal{B}$ responds as follows when $\mathcal{A}$ issues the decryption query $C_i = (U, V_1, \ldots, V_n, W_1, W_2, \mathcal{L}, \tau)$. $\mathcal{B}$ checks if the tuple (q_i, τ) is on the H_3-list. If it is on the list, sets $g = R$ which is the first term of q_i. Otherwise, pick a random $g \in \mathcal{F}$. $\mathcal{B}$ then checks if (g, h_i) is on the H_2-list. If h_i appears on the list, $\mathcal{B}$ computes $m = W_2 \oplus h_i$. Otherwise, $\mathcal{B}$ picks $h_i \in \{0,1\}^{k_1}$ randomly, add (g, h_i) to the H_2-list, and computes $m = W_2 \oplus h_i$. If the (q_i, τ) was not on the H_3-list, add $((R, m, U, V_1, \ldots, V_n, W_1, W_2, \mathcal{L}), \tau)$ to the H_3-list. $\mathcal{B}$ responds m as the decryption of C_i.

$\mathcal{A}$'s view of the signature is identical to that of a real signature, and thus its probability of success in breaking the encryption scheme is unchanged, $\epsilon' = \epsilon$. □

I Proof of Proposition 23

Proof. This proof is strait-forward. We prove this security by the reduction.

Let $\mathcal{A}$ an adversary that attacks the MID-IND-EUF security of `MIDComb`. $\mathcal{A}$ breaks the indistinguishability or the existential unforgeability of `MIDComb`. From Lemma 21 and Lemma 22, we can construct an attacker $\mathcal{A}'$ that breaks the indistinguishability of `MIDEnc` alone or $\mathcal{A}''$ that breaks the existential unforgeability of `IDSig` alone with non-negligible probability. The security of `MIDEnc` is proved by Baek et al. [2], and by using $\mathcal{A}'$, the GBDH problem can be solved. The security of `IDSig` is proved by Cha and Cheon [15]. By using $\mathcal{A}''$, the CDH problem in $\mathcal{G}$ can be

solved. Which means that on inputs of (P, aP, bP), abP can be solved. The answer of the GBDH problem is computed by $\hat{e}(P, P)^{abc} = \hat{e}(abP, cP)$. □