Des. Codes Cryptogr. (2016) 79:63–85
DOI 10.1007/s10623-015-0035-0

An efficient IBE scheme with tight security reduction in the random oracle model

Jong Hwan Park · Dong Hoon Lee

Received: 3 June 2014 / Revised: 25 October 2014 / Accepted: 6 January 2015 /
Published online: 18 January 2015

Abstract We present a new practical identity-based encryption (IBE) system that can be another candidate for standard IBE techniques. Our construction is based on a new framework for realizing an IBE trapdoor from pairing-based groups, which is motivated from the 'two equation' revocation technique suggested by Lewko et al. (IEEE Symposium on Security and Privacy, 2010). The new framework enables our IBE system to achieve a tight security reduction to the Decisional Bilinear Diffie–Hellman assumption in the random oracle model. Due to its the tightness, our system can take as input the shorter size of security parameters than the previous practical BF, SK, and BB_1 systems, which provides better efficiency to our system in terms of computational cost.

Keywords Identity based encryption · Bilinear maps · Tight reduction

Mathematics Subject Classification 68P25 · 94A60

1 Introduction

Identity-based encryption (IBE) [45] is a special type of public key encryption where a public key can be any string that carries its own meaning to a user's identity, such as an e-mail address. As such a meaningful string can be naturally associated with a user, an IBE system does not need a certifying mechanism to ensure that a public key (as the meaningful string) is bound to a user (as the owner of the public key). As opposed to an IBE system, a traditional public

Communicated by L. Perret.

J. H. Park
Department of Computer Science, Sangmyung University, Seoul, Korea
e-mail: jhpark@smu.ac.kr

D. H. Lee (✉)
Graduate School of Information Security, Korea University, Seoul, Korea
e-mail: donghlee@korea.ac.kr

key encryption system needs the certifying mechanism to securely distribute public keys, and indeed it must run on a complex architecture called 'public key infrastructure'.

By virtue of the advantage over the public key encryption, IBE had received considerable interest to cryptographic researchers since Shamir [45] posed the initial question about the existence of such an IBE system. In 2001, Boneh and Franklin [13] proposed the first practical IBE system based on groups with efficiently computable bilinear maps (i.e., paring), along with a formal security definition of IBE. Since then, a large body of work [10,20–22,29, 38,39,43,46,47] has been devoted to constructing pairing-based IBE systems to improve in terms of security and efficiency. Among the previous IBE systems, three of them have been perceived as practical constructions, which are works by Boneh and Franklin [13], Sakai and Kasahara [20,22,43], and Boneh and Boyen [10] (denoted as 'BB_1'), and thereafter they have been submitted to the IEEE P1363.3 standard for "Identity-Based Cryptographic Techniques using Pairings".

1.1 Our contribution

In this work we present a new practical IBE system that can be another candidate for standard IBE techniques. Our IBE system results from a new framework for realizing the IBE trapdoor, which is motivated by the 'two equation' technique recently suggested by Lewko et al. [40]. One notable advantage of the new framework is that our construction is also pairing-based like BF, SK, and BB_1 systems, yet it has a *tight* security reduction to the standard decision bilinear Diffie–Hellman (DBDH) assumption. In order to encrypt arbitrary-length messages, we also suggest a new identity-based key encapsulation mechanism (IBKEM) combined with one-time symmetric-key encryption. Our IBE systems are all proven to be fully secure against chosen ciphertext attacks in the random oracle model. In particular, one-time symmetric-key encryption secure against passive attacks is sufficient for the latter IBE system without the need of the 'encrypt-then-MAC' or 'authenticated symmetric encryption' paradigm.

None of the three practical BF, SK, and BB_1 systems provided tightness in their respective security analysis, and in fact there existed significant security gaps between security assumptions and their IBE systems. Later, Attrapadung et al. [5] presented a variant of the BF system that is tightly secure under the DBDH assumption, using the Katz and Wang [33,36] key generation technique. One might wonder what the benefit from such tightness in security reduction is. The benefit is that we can achieve security of our system straightforwardly from that of the underlying DBDH assumption at the same security level. This means that if we want to instantiate our IBE system at current 80-bit security level, we can use a DBDH-hard group *at the same security level*. However, this is not the case in BF, SK, and BB_1 systems where security is loosely reduced to each security assumption by a factor of (at least) 2^{50} if we consider a reasonable number of adversarial hash queries as 2^{50}. The loose security reduction forces us to choose a larger security parameter (regarding DBDH-hard groups) than the 80-bit one even if we want to instantiate them at the 80-bit security level. Importantly, the larger security parameter tends to have an unfavorable effect on computational cost of resulting IBE systems [16]. For instance, when comparing BF, SK, and BB_1 systems at 128-bit security level with our system at 80-bit security level, ours has about at least 10 times faster decryption than the three systems, and about 22 times faster encryption than the BF system. Also, when compared to Attrapadung et al.'s IBE system (denoted by 'AFG^+') at 80-bit same security level, ours requires 1.2 times longer size of ciphertext, but instead ours achieves about 8.6 times faster encryption and 1.6 times faster decryption under groups with symmetric bilinear maps. To add credence to these results, we give more concrete performance comparison in terms of security and efficiency in Sect. 5.

1.2 Overview of our new technique

BF, SK, and BB_1 systems have their unique frameworks to realize IBE trapdoors from paring-based groups, respectively. Following Boyen's naming in [16], each framework is called 'full-domain-hash' (for BF), 'exponent-inversion' (for SK), and 'commutative-blinding' (for BB_1). Each framework determines both the distinct structure of a private key and different kinds of security assumptions. Also, most of the subsequent paring-based IBE systems fall into one of the three paradigms.

To build our new IBE system, we also come up with a new framework for realizing the IBE trapdoor. As we mentioned before, our framework is motivated by the two equation technique of Lewko et al. [40]. Roughly speaking, the original LSW technique is to use private key elements $(g^r, (g_1 u^{\mathsf{ID}})^r)$ and ciphertext elements $(g^s, (g_1 u^{\mathsf{ID}'})^s)$ to compute a pairing value $e(g,u)^{sr}$. Here, g, g_1, and u are from public parameters. The value $e(g,u)^{sr}$ is then used to recover a message blinding factor $e(g,g)^{\alpha s}$ by pairing g^s with an additional key element $g^\alpha u^r$. The point is that such a pairing value can be obtained only if $\mathsf{ID} \neq \mathsf{ID}'$, and this gives the revocation system [40] where only users whose identities are different from ID' are able to compute the pairing value. On the other hand, our two equation technique is slightly changed in a way that computing the value $e(g,u)^{sr}$ is possible only if $\mathsf{ID} = \mathsf{ID}'$. This can be done by setting private key elements as $(g^r, (\mathcal{H}(\mathsf{ID})u^{\mathrm{tag}_k})^r)$ and ciphertext elements as $(g^s, (\mathcal{H}(\mathsf{ID}')u^{\mathrm{tag}_c})^s)$, where $\mathcal{H}$ is a cryptographic hash function and (as we explain below) the probability that $\mathrm{tag}_k = \mathrm{tag}_c$ is negligible.

As in the BF system, our framework requires a cryptographic hash function $\mathcal{H}$ that maps an identity string $\mathsf{ID} \in \{0,1\}^*$ to a group element $\mathcal{H}(\mathsf{ID})$, but unlike the BF system a private key for an identity ID is not uniquely determined. A private key $\mathsf{sk}_{\mathsf{ID}}$ consists of three groups $(g^\alpha u^r, g^r, (\mathcal{H}(\mathsf{ID})u^{\mathrm{tag}_k})^r)$, which differs by a randomly-chosen exponent r in $\mathbb{Z}_p$. Here, α is the master secret key known only to a key generation center. At this moment, one may wonder how the value tag_k is decided. Indeed, we have that $\mathrm{tag}_k = h(g^\alpha u^r, g^r) \in \mathbb{Z}_p$ by introducing another cryptographic hash function h. Similarly, when encrypting a message M, a ciphertext under ID is constructed as $\left(Me(g,g)^{\alpha s}, g^s, (\mathcal{H}(\mathsf{ID})u^{\mathrm{tag}_c})^s\right)$ using the hash value $\mathrm{tag}_c = h(Me(g,g)^{\alpha s}, g^s)$. In case of our IBKEM, an arbitrary length message M is encrypted as $\left(\mathcal{E}_K(M), g^s, (\mathcal{H}(\mathsf{ID})u^{\mathrm{tag}_c})^s\right)$ using a one-time symmetric-key encryption algorithm $\mathcal{E}$, where $K = e(g,g)^{\alpha s}$ and $\mathrm{tag}_c = h(\mathcal{E}_K(M), g^s)$. If we use a collision-resistant hash function h, the correctness error caused by the equality $\mathrm{tag}_k = \mathrm{tag}_c$ becomes acceptable in practice. Another characteristic of our framework is to use the hash function h to protect the ciphertext element $Me(g,g)^{\alpha s}$ or $\mathcal{E}_K(M)$ related to M. Indeed, the distinct usage of h enables our system to directly obtain chosen ciphertext security without resorting to other methods such as 'encrypt-then-MAC' or 'authenticated symmetric encryption' or Fujisaki-Okamoto transform [27].

We now explain how our IBE system can achieve a tight security reduction under the DBDH assumption. In our security proofs, the two hash functions $\mathcal{H}$ and h are modeled as random oracles. Somewhat surprisingly, being able to use two hash functions in generating one group element enables our reduction algorithm to generate private keys for *all* identities and use *any* identity as a challenge identity ID^*. Nevertheless, a private key for ID^* is not helpful to decrypt the challenge ciphertext (that can be constructed under ID^*), which is necessary for solving the so-called '*self-decryption*' paradox. Notice that similar reductions can be found in [29,47] that provided full security without random oracles. Let (g, g^a, g^b, g^c, T) be a DBDH instance. Given an identity ID_i, the random oracle $\mathcal{H}$ outputs $\mathcal{H}(\mathsf{ID}_i) = (g^a)^{\gamma_i} g^{\pi_i}$ for two randomly-chosen exponents γ_i and π_i in $\mathbb{Z}_p$. The important point is that the value γ_i

per each identity can be information-theoretically hidden from an adversary's view and later used for an output value of another random oracle h. When creating a private key for ID_i, our reduction algorithm is able to generate the key as $\left(g^{\alpha}u^{\widetilde{r}}, g^{\widetilde{r}}, (\mathcal{H}(\mathsf{ID}_i)u^{\mathrm{tag}_k})^{\widetilde{r}}\right)$ by setting $\mathrm{tag}_k = h(g^{\alpha}u^{\widetilde{r}}, g^{\widetilde{r}}) = \gamma_i$ and $\widetilde{r} = b + r$ for a random r in $\mathbb{Z}_p$. The validity of the private key is checked under the condition that $\alpha = ab$ and $u = g^{-a}g^{\delta}$ for a random $\delta \in \mathbb{Z}_p$. A similar manipulation is taken when generating the challenge ciphertext under ID^*. As $\mathsf{sk}_{\mathsf{ID}^*}$ must not be asked, the value γ^* embedded into $\mathcal{H}(\mathsf{ID}^*)$ will be hidden until the challenge phase (with overwhelming probability) and thus can be reserved for setting $\mathrm{tag}_c = \gamma^*$ (as well as $s = c$ for the DBDH problem). In case when trying to decrypt the challenge ciphertext using $\mathsf{sk}_{\mathsf{ID}^*}$, it should be the case that $\mathrm{tag}_k = \gamma^*$ that was already embedded into $\mathcal{H}(\mathsf{ID}^*)$. Therefore, the decryption is not possible because $\mathrm{tag}_k = \mathrm{tag}_c$, and this explains how the self-decryption paradox can be solved.

To achieve the chosen ciphertext security, our reduction algorithm needs to deal with adversarial decryption queries. In our security analysis, this is not a big problem as private keys for all identities can be generated and ill-formed ciphertexts are detected via consistency check using pairing. The only problem is that in the event that $\mathrm{tag}_k = \mathrm{tag}_c$ happens, normal decryption cannot be performed. However, as an output of h as a random oracle is determined by choosing a random value in $\mathbb{Z}_p$ and p is exponentially large (e.g., p is a 160-bit prime), our reduction can avoid such a troublesome case in all decryption queries with overwhelming probability.

1.2.1 Comparison to the Katz–Wang technique

As in the BF system and ours, the Katz–Wang technique is based on a cryptographic hash function $\mathcal{H}$ that maps an identity string $\mathsf{ID} \in \{0,1\}^*$ to a group element $\mathcal{H}(\mathsf{ID})$ is required. The key idea is that two public keys $\mathcal{H}(\mathsf{ID}_i, 0)$ and $\mathcal{H}(\mathsf{ID}_i, 1)$ both are used in encrypting a message for one identity ID_i, but a private key corresponding to one of two public keys is given to a user with ID_i. In security analysis, $\mathcal{H}(\mathsf{ID}_i, b)$ is programmed to extract a private key for a randomly chosen $b \in \{0,1\}$, whereas the other $\mathcal{H}(\mathsf{ID}_i, \overline{b})$ is programmed to calculate a CDH value. Therefore, a reduction algorithm can only create a private key for $\mathcal{H}(\mathsf{ID}_i, b)$, which is enough to answer a private key query. If ID_i is chosen as the challenge identity, the other $\mathcal{H}(\mathsf{ID}_i, \overline{b})$ will be used to deal with the DBDH problem. This shows that the reduction algorithm can answer private key queries for *all* identities and use *any* identity as the challenge ID^*. The self-decryption paradox is then resolved from the fact that the reduction algorithm cannot generate a private key for $\mathcal{H}(\mathsf{ID}_i, \overline{b})$ itself.

Compared to our new framework, the Katz–Wang technique causes inefficiency in terms of encryption and decryption costs. In case of the AFG$^+$ system, for instance, two paring computations corresponding to both $\mathcal{H}(\mathsf{ID}_i, 0)$ and $\mathcal{H}(\mathsf{ID}_i, 1)$ should be performed in every encryption, which makes encryption cost a lot more expensive than ours. Moreover, decryption algorithm requires more computation to determine whether ciphertext elements corresponding to $\mathcal{H}(\mathsf{ID}_i, \overline{b})$ is well formed. This is because each user is given a private key for one $\mathcal{H}(\mathsf{ID}_i, b)$ for a random $b \in \{0,1\}$ and thus ciphertext elements for $\mathcal{H}(\mathsf{ID}_i, \overline{b})$ cannot be decrypted directly. In case of the AFG$^+$ system, a variant of Fujisaki–Okamoto transform is used, so that decrypting ciphertext elements for $\mathcal{H}(\mathsf{ID}_i, b)$ can yield a random value and message that are then used to re-generate ciphertext elements corresponding to $\mathcal{H}(\mathsf{ID}_i, \overline{b})$.

1.2.2 Comparison to the Coron technique

A similar technique to our key generation framework was previously used by Coron [24], where a private key for ID is generated as $((\mathcal{H}(\mathsf{ID})u^{-y})^a, y)$ using the (fixed) master secret key a and a newly chosen random y. The idea behind their security proof was also similar to ours by manipulating the hash output $\mathcal{H}(\mathsf{ID})$ as $(g^a)^y g^r$ for two randomly-chosen y and r, which leads to a tight security reduction. However, there are several differences between them: in terms of security, Coron-IBE system relies on the square-DBDH assumption[1] [37] whereas our system relies on the DBDH assumption. Obviously, the square-DBDH assumption [37] is *stronger* than the standard DBDH assumption. In terms of efficiency, Coron-IBE system provides faster decryption than ours, but instead slower encryption and longer size of ciphertexts than ours when comparing both schemes at the same security level. We notice that a variant of Coron-IBE system was also suggested in [24], which achieves a tight security reduction under the DBDH assumption. As the variant provides almost the same efficiency as the original Coron's system, it has slower encryption and faster decryption, compared to ours. In Sect. 5, we will give an efficiency comparison between the variant and ours in more detail.

1.3 Related work

Boneh and Franklin [13] presented the first practical IBE system based on groups with efficiently computable pairings and defined the formal security notion for IBE known as full security against chosen ciphertext attacks. Most of the subsequent IBE systems followed the notion depending on different kinds of security assumptions. Until now, BF [13], SK [20,22,43], and BB_1 [10] systems have been considered as practical constructions and their security was all demonstrated in the random oracle model.

In an attempt to prove security without random oracles, Canetti et al. [18] suggested a weaker security notion for IBE, known as selective-ID security. Following the weaker notion, Boneh and Boyen [10] proposed two efficient IBE systems, one of which was the basis for BB_1. Many IBE systems [21,29,38,39,46,47] were later suggested to achieve full security without random oracles, but they all become less efficient than the random oracle constructions in practical aspects such as public parameter size or achieving chosen ciphertext security.

Regarding tight security reduction, Attrapadung et al. [5] proposed a Katz–Wang variant of the BF system whose security is tightly reduced to the DBDH assumption. Their construction is fully secure against chosen ciphertext attacks in the random oracle model, but less efficient than our IBE system in terms of both encryption and decryption costs (see Sect. 5 for concrete performance comparison). On the other hand, Gentry IBE [29] achieved the full security without random oracles. Tightness in its security reduction was achieved by relying on a (non-standard) q-type assumption where q depends on the number of private key queries that an adversary makes.

The notion of IBE has been extended in two flavors. In a vertical (and hierarchical) extension, IBE can provide a 'delegation' mechanism [31,35] by which private keys for lower-level identities are created from an upper-level identity but the reverse is not possible. Many works [10,11,17,30,31,44,46,47] have been suggested to realize such a delegation mechanism, also known as hierarchical IBE (HIBE). In a horizontal extension, IBE becomes the special case

[1] The square-DBDH assumption is defined from the following problem: given (g, g^a, g^b, T) as input, determine $T = e(g, g)^{a^2 b}$ or random.

of the attribute-based encryption (ABE) [9,34,42], where attributes (instead of a single identity) are associated with private keys and ciphertexts, respectively, and decryption works only if attributes satisfy a function depending on each ABE system. Furthermore, when attributes (an identity) embedded into ciphertexts are encrypted, ABE (IBE) can also be extended for providing searchable techniques [1,12] on encrypted data. Recently, the horizontal extensions are conceptually united under the notion of functional encryption (FE) [15].

Finally, we notice that there exist other approaches to build IBE trapdoors without pairings. Cocks [23] and Boneh et al. [14] constructed IBE systems based on the quadratic-residuosity problem and Gentry et al. [32] demonstrated how to build an IBE system based on lattice. Recently, lattice-based IBE can also be extended toward HIBE [2,3,19] and FE [4] constructions.

2 Preliminaries

2.1 Identity-based encryption

An IBE system consists of four algorithms:

- **Setup**(k) takes a security parameter k as input and outputs a public parameter PP and a master secret key msk.
- **KeyGen**(msk, PP, ID) takes a master secret key msk, a public parameter PP and an identity $\mathsf{ID} \in \mathcal{ID}$ as inputs, where $\mathcal{ID}$ is an identity space. It outputs $\mathsf{sk}_{\mathsf{ID}}$, a private key for ID.
- **Encrypt**(PP, M, ID) takes a public parameter PP, a message $M \in \mathcal{M}$, and an identity $\mathsf{ID} \in \mathcal{ID}$ as inputs, where $\mathcal{M}$ is a message space. It outputs CT under ID, a ciphertext under ID.
- **Decrypt**(CT, PP, $\mathsf{sk}_{\mathsf{ID}}$) takes a ciphertext CT under ID, a public parameter PP, and a private key $\mathsf{sk}_{\mathsf{ID}}$ as inputs. It outputs a message M or a random message.

Correctness For all $\mathsf{ID} \in \mathcal{ID}$ and all $M \in \mathcal{M}$, let $(\mathsf{PP}, \mathsf{msk}) \leftarrow$ **Setup**(k), $\mathsf{sk}_{\mathsf{ID}} \leftarrow$ **KeyGen**(msk, PP, ID), $\mathsf{CT} \leftarrow$ **Encrypt**(PP, M, ID). We have $M \leftarrow$ **Decrypt**($\mathsf{sk}_{\mathsf{ID}}$, PP, CT).

We next define chosen ciphertext security [13] of an IBE system. The security is defined via the following game interacted by a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$:

- **Setup**: $\mathcal{C}$ runs the setup algorithm to obtain a public parameter PP and a master secret key msk. $\mathcal{C}$ gives PP to $\mathcal{A}$.
- **Query Phase 1**: $\mathcal{A}$ adaptively issues a number of queries where each query is one of:
  - Private key query on ID: $\mathcal{C}$ runs the key generation algorithm to obtain a private key for ID and gives the key $\mathsf{sk}_{\mathsf{ID}}$ to $\mathcal{A}$.
  - Decryption query on (CT, ID): $\mathcal{C}$ runs the key generation algorithm to obtain $\mathsf{sk}_{\mathsf{ID}}$ to $\mathcal{A}$ and then runs the decryption algorithm using $\mathsf{CT}_{\mathsf{ID}}$ and $\mathsf{sk}_{\mathsf{ID}}$. It gives the resulting message to $\mathcal{A}$.
- **Challenge**: $\mathcal{A}$ outputs two equal-length messages M_0, M_1 and an identity ID^* on which it wishes to be challenged. The only restriction is that ID^* is not queried in Query Phase 1. $\mathcal{C}$ flips a coin $\sigma \in \{0, 1\}$. $\mathcal{C}$ gives $\mathsf{CT}^* \leftarrow$ **Encrypt**(PP, M_σ, ID^*) as a challenge ciphertext to $\mathcal{A}$.
- **Query Phase 2**: $\mathcal{A}$ adaptively issues a number of additional queries where each query is one of:

- Private key query on ID, where $\mathsf{ID} \neq \mathsf{ID}^*$: $\mathcal{C}$ responds as in Query Phase 1.
- Decryption query on $(\mathsf{CT}, \mathsf{ID})$, where $(\mathsf{CT}, \mathsf{ID}) \neq (\mathsf{CT}^*, \mathsf{ID}^*)$: $\mathcal{C}$ responds as in Query Phase 1.

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

The advantage of the adversary $\mathcal{A}$ in breaking the chosen ciphertext security of an IBE system $\mathcal{IBE}$ is defined as $\mathbf{Adv}^{\mathrm{CCA}}_{\mathcal{IBE},\mathcal{A}} = \left|\Pr[b' = b] - 1/2\right|$.

Definition 1 We say that an IBE system is (t, ϵ, q_K, q_D)-IND-ID-CCA secure if $\mathbf{Adv}^{\mathrm{CCA}}_{\mathcal{IBE},\mathcal{A}} < \epsilon$ for any adversary $\mathcal{A}$ that runs in time at most t, issues at most q_K private key queries, and issues at most q_D decryption queries in chosen ciphertext security games.

2.2 One-time symmetric-key encryption

A one-time symmetric-key encryption scheme consists of two algorithms: a deterministic encryption algorithm $\mathcal{E}$ takes a message $M \in \{0, 1\}^*$ and a key $K \in \mathcal{K}$ as inputs and outputs a ciphertext $C = \mathcal{E}_K(M)$. Here, $\mathcal{K}$ is a key space that is determined by a security parameter $k \in \mathbb{Z}^+$. Another deterministic algorithm $\mathcal{D}$ is a decryption algorithm that takes a ciphertext C and a key K as inputs and outputs a message $M = \mathcal{D}_K(C)$.

We define security for a one-time symmetric-key encryption scheme $\mathcal{SKE} = (\mathcal{E}, \mathcal{D})$, which is security against *passive* attacks [25]. The security is defined via the following game interacted by a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$:

– **Setup**: $\mathcal{C}$ chooses a random key K in key space $\mathcal{K}(k)$.
– **Challenge**: $\mathcal{A}$ outputs two equal-length messages M_0 and M_1. $\mathcal{C}$ flips a coin $\sigma \in \{0, 1\}$ and gives $C^* \leftarrow \mathcal{E}_K(M_\sigma)$ as a challenge ciphertext to $\mathcal{A}$.
– **Guess**: $\mathcal{A}$ outputs a guess $\sigma' \in \{0, 1\}$. $\mathcal{A}$ wins if $\sigma' = \sigma$.

The advantage of the adversary $\mathcal{A}$ in breaking the passive security of a one-time symmetric-key encryption scheme $\mathcal{SKE}$ is defined as $\mathbf{Adv}^{\mathrm{OT\text{-}IND}}_{\mathcal{SKE},\mathcal{A}} = \left|\Pr[b' = b] - 1/2\right|$.

Definition 2 We say that a one-time symmetric-key encryption scheme is (t, ϵ)-secure against passive attacks if $\mathbf{Adv}^{\mathrm{OT\text{-}IND}}_{\mathcal{SKE},\mathcal{A}} < \epsilon$ for any adversary $\mathcal{A}$ that runs in time at most t in passive attack games.

2.3 Bilinear maps and complexity assumptions

2.3.1 Bilinear maps

Our schemes will be parameterized by a pairing parameter generator. This is an algorithm $\mathcal{G}$ that on input $k \in \mathbb{N}$ returns the description of multiplicative cyclic groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of prime order p, the description of a multiplicative cyclic group $\mathbb{G}_T$ of the same order, and a non-degenerate bilinear pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. We assume that g_1 is a generator of $\mathbb{G}_1$ and g_2 is a generator of $\mathbb{G}_2$. Following the standard notation in [10,13], we assume that the function e has the following three properties.

1. Bilinear: for all $u \in \mathbb{G}_1$, $v \in \mathbb{G}_2$ and $a, b \in \mathbb{Z}$, we have $e(u^a, v^b) = e(u, v)^{ab}$.
2. Non-degenerate: $e(g_1, g_2) \neq 1$.
3. Computable: there is an efficient algorithm to compute the map e.

2.3.2 The asymmetric decisional bilinear Diffie–Hellman (DBDH) problem

The asymmetric DBDH problem [41] is defined as follows: given $(g_1, g_1^a, g_1^b, g_1^c, g_2, g_2^a, g_2^b, g_2^c, T) \in \mathbb{G}_1^4 \times \mathbb{G}_2^4 \times \mathbb{G}_T$ as input, output 1 if $T = e(g_1, g_2)^{abc}$ and 0 otherwise. For our security analysis, it suffices to consider a slightly weaker problem that does not need to take g_1^a as input. Hereafter, we refer to the asymmetric DBDH problem as the weaker one. We say that an algorithm $\mathcal{A}$ that outputs $\sigma \in \{0, 1\}$ has an advantage $\mathbf{Adv}^{\mathrm{ADBDH}}_{\mathbb{G},\mathcal{A}} = \epsilon$ in solving the asymmetric DBDH problem in $\mathbb{G}$ if

$$\Big|\Pr\big[\mathcal{A}(g_1, g_1^b, g_1^c, g_2, g_2^a, g_2^b, g_2^c, e(g_1, g_2)^{abc}) = 0\big] - \Pr\big[\mathcal{A}(g_1, g_1^b, g_1^c, g_2, g_2^a, g_2^b, g_2^c, R) = 0\big]\Big| \geq \epsilon,$$

where the probability is taken over the random choice of $a, b, c \in \mathbb{Z}_p$, the random choice of $R \in \mathbb{G}_T$, and the random bits used by $\mathcal{A}$.

Definition 3 We say that the (t, ϵ)-asymmetric DBDH assumption holds in $\mathbb{G}_1$ and $\mathbb{G}_2$ if no polynomial time adversary $\mathcal{A}$ that runs in time at most t has at least advantage ϵ in solving the asymmetric DBDH problem in $\mathbb{G}_1$ and $\mathbb{G}_2$.

3 Our IBE system

3.1 Construction

Setup(k)**:** Given a security parameter $k \in \mathbb{Z}^+$, the setup algorithm runs $\mathcal{G}(k)$ to obtain a tuple $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$. The algorithm selects a random generator $g_1 \in \mathbb{G}_1$, a random generator $g_2 \in \mathbb{G}_2$, a random group element $u \in \mathbb{G}_2$, and a random exponent $\alpha \in \mathbb{Z}_p$. The algorithm sets $A = e(g_1, g_2)^\alpha$ and chooses two cryptographic hash functions $H_1 : \{0, 1\}^* \to \mathbb{G}_2$ and $H_2 : \{0, 1\}^* \to \mathbb{Z}_p$. The public parameters PP (with the description of $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$) and the master secret key msk are generated as

$$\mathsf{PP} = \big(g_1, u, A, H_1, H_2\big), \quad \mathsf{msk} = g_2^\alpha.$$

KeyGen$(\mathsf{msk}, \mathsf{PP}, \mathsf{ID})$**:** To create a private key $\mathsf{sk}_{\mathsf{ID}}$ for an identity $\mathsf{ID} \in \mathcal{ID}$, the key generation algorithm does the following:

1. Pick a random exponent $r \in \mathbb{Z}_p$.
2. Compute $d_0 = g_2^\alpha u^r \in \mathbb{G}_2$, $d_1 = g_1^r \in \mathbb{G}_1$, and $\mathrm{tag}_k = H_2(d_0, d_1) \in \mathbb{Z}_p$.
3. Compute $d_2 = \big(H_1(\mathsf{ID})u^{\mathrm{tag}_k}\big)^r \in \mathbb{G}_2$.
4. Output a private key $\mathsf{sk}_{\mathsf{ID}} = (d_0, d_1, d_2, \mathrm{tag}_k) \in \mathbb{G}_2 \times \mathbb{G}_1 \times \mathbb{G}_2 \times \mathbb{Z}_p$.

Encrypt$(\mathsf{PP}, \mathsf{ID}, M)$**:** To encrypt a message $M \in \mathbb{G}_T$ under an identity $\mathsf{ID} \in \mathcal{ID}$, the encryption algorithm does as follows:

1. Pick a random exponent $s \in \mathbb{Z}_p$.
2. Compute $C_0 = A^s M \in \mathbb{G}_T$, $C_1 = g_1^s \in \mathbb{G}_1$, and $\mathrm{tag}_c = H_2(C_0, C_1) \in \mathbb{Z}_p$.
3. Compute $C_2 = \big(H_1(\mathsf{ID})u^{\mathrm{tag}_c}\big)^s \in \mathbb{G}_2$.
4. Output a ciphertext $\mathsf{CT} = (C_0, C_1, C_2) \in \mathbb{G}_T \times \mathbb{G}_1 \times \mathbb{G}_2$.

Decrypt$(\mathsf{PP}, \mathsf{CT}, \mathsf{sk}_{\mathsf{ID}})$**:** To decrypt a ciphertext (C_0, C_1, C_2) using a private key $\mathsf{sk}_{\mathsf{ID}} = (d_0, d_1, d_2, \mathrm{tag}_k)$ for ID, the decryption algorithm does as follows:

1. Compute $\mathrm{tag}_c = H_2(C_0, C_1) \in \mathbb{Z}_p$ and check if $e\big(C_1, H_1(\mathsf{ID})u^{\mathrm{tag}_c}\big) \stackrel{?}{=} e(g_1, C_2)$.
2. If the equality above fails, output a random $\widehat{M} \in \mathbb{G}_T$.
3. Otherwise, check if $\mathrm{tag}_c \stackrel{?}{=} \mathrm{tag}_k$ in $\mathbb{Z}_p$.
4. If the equality above holds, output $\perp$.
5. Otherwise, compute

$$M = C_0 \Big/ e(C_1, d_0) \cdot \left(\frac{e(d_1, C_2)}{e(C_1, d_2)}\right)^{\frac{-1}{\mathrm{tag}_c - \mathrm{tag}_k}}. \tag{1}$$

3.1.1 Correctness

If $\mathrm{tag}_c = \mathrm{tag}_k$ in $\mathbb{Z}_p$, the decryption algorithm does not work, so that it has the correctness error $1/p$ on each decryption. Otherwise, we can verify that the decryption algorithm works correctly for well-formed ciphertexts as follows:

$$\begin{aligned} e(C_1, d_0) \cdot \left(\frac{e(d_1, C_2)}{e(C_1, d_2)}\right)^{\frac{-1}{\mathrm{tag}_c - \mathrm{tag}_k}} &= e\big(g_1^s, g_2^\alpha u^r\big) \cdot \left(\frac{e\big(g_1^r, (H_1(\mathsf{ID})u^{\mathrm{tag}_c})^s\big)}{e\big(g_1^s, (H_1(\mathsf{ID})u^{\mathrm{tag}_k})^r\big)}\right)^{\frac{-1}{\mathrm{tag}_c - \mathrm{tag}_k}} \\ &= e(g_1, g_2^\alpha)^s e(g_1, u)^{sr} \cdot e\big(g_1^{sr}, u^{\mathrm{tag}_c - \mathrm{tag}_k}\big)^{\frac{-1}{\mathrm{tag}_c - \mathrm{tag}_k}} \\ &= A^s. \end{aligned}$$

3.1.2 Encryption and decryption costs

In encryption, the three exponentiations A^s, g_1^s, and $u^{\mathrm{tag}_c \cdot s}$ can be calculated in fixed bases $A \in \mathbb{G}_T$, $g_1 \in \mathbb{G}_1$, and $u \in \mathbb{G}_2$, respectively. Instead, the hashing $H_1(\mathsf{ID})$ and its exponentiation $H_1(\mathsf{ID})^s$ in $\mathbb{G}_2$ will be done separately without precomputation in usual situations. Thus, the encryption cost becomes one fixed-base exponentiation in each group $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$, respectively, one hashing into $\mathbb{G}_2$, and one general exponentiation in $\mathbb{G}_2$.

Upon decryption, it seems that the decryption algorithm requires computing five pairings, but these can be saved into two parings. We first can change the above formula (1) into:

$$e(C_1, d_0) \cdot \left(\frac{e(d_1, C_2)}{e(C_1, d_2)}\right)^{\frac{-1}{\mathrm{tag}_c - \mathrm{tag}_k}} = \frac{e\Big(d_1^{\frac{-1}{\mathrm{tag}_c - \mathrm{tag}_k}}, C_2\Big)}{e\Big(C_1, d_0^{-1} d_2^{\frac{-1}{\mathrm{tag}_c - \mathrm{tag}_k}}\Big)}.$$

Next, by using the *implicit* consistency check [38], we do not need to perform the pairing consistency test explicitly. Instead, the decryption algorithm randomizes two elements $H_1(\mathsf{ID})u^{\mathrm{tag}_c}$ and g_1 by raising a randomly chosen exponent $\widetilde{r} \in \mathbb{Z}_p$ and performs the following computation:

$$\frac{e\Big(d_1^{\frac{-1}{\mathrm{tag}_c - \mathrm{tag}_k}} g_1^{\widetilde{r}}, C_2\Big)}{e\Big(C_1, d_0^{-1} d_2^{\frac{-1}{\mathrm{tag}_c - \mathrm{tag}_k}} (H_1(\mathsf{ID})u^{\mathrm{tag}_c})^{\widetilde{r}}\Big)}. \tag{2}$$

If the pairing test passes, the output of the above equation becomes the same as that of the original decryption algorithm. Otherwise, the fresh random value $\widetilde{r}$ chosen by the decryption algorithm survives and thus prevents an adversary from gaining any information on an ill-formed ciphertext. As a consequence, the decryption cost is determined by the computation

in the Eq. 2 that shows five exponentiations and two pairing computations. All the exponentiations can be done in fixed bases such as $g_1, d_1 \in \mathbb{G}_1$ and $d_2, u, H_1(\mathsf{ID}) \in \mathbb{G}_2$. Notice that a user with identity ID can compute $H_1(\mathsf{ID}) \in \mathbb{G}$ in advance and prepare for fixed bases related to it, regardless of any received ciphertext. Thus, the decryption cost is concluded with two fixed-base exponentiations in $\mathbb{G}_1$, three fixed-base exponentiations in $\mathbb{G}_2$, and two parings.

3.1.3 Achieving perfect correctness

Upon decryption, our decryption algorithm cannot proceed in the event that $\text{tag}_c = \text{tag}_k$ occurs. Obviously, the probability that the event happens is negligible when the value tag is in $\mathbb{Z}_p$ and p is represented by approximately 160 bits. However, in order to avoid even the negligible correctness error, we can employ an approach suggested in a recent tag-based dual system encryption [47]. A possible solution is to simply run an efficient selectively (chosen-ciphertext) secure IBE system [10] in parallel. When a message is encrypted under ID with tag_c, an encryptor also encrypts the message under the tag_c (as an identity) in the second selective system. When $\text{tag}_c \neq \text{tag}_k$, we can use our original IBE system. In the unlikely event that $\text{tag}_c = \text{tag}_k$, we can use the second ciphertext. An alternative approach in [47] such as giving two private keys for an identity ID seems to not be applicable to our system, because our security analysis shows that a hash value $H_1(\mathsf{ID})$ should be assigned to only one tag_k.

3.1.4 Stateful key generation algorithm

According to our security analysis, the **KeyGen** algorithm should be stateful, in that the algorithm stores a random exponent $r \in \mathbb{Z}_p$ used to generate a private key for an identity and later the exponent should be used again when the same identity is requested for a private key generation.

3.2 Security

Theorem 1 *Let H_1 and H_2 be modeled as random oracles. Suppose the (t', ϵ')-asymmetric DBDH assumption holds in $\mathbb{G}_1$ and $\mathbb{G}_2$. Then our IBE system is (t, ϵ, q_K, q_D)-IND-ID-CCA secure, where*

$$\epsilon \leq \left(1 - \frac{q_{H_2}}{p} - \frac{2q_D}{p}\right) \cdot \epsilon',$$
$$t \geq t' - \mathcal{O}((q_K + q_D + q_{H_1}) \cdot t_e) - \mathcal{O}(q_D \cdot t_p).$$

Here, $\{q_{H_1}, q_{H_2}\}$ is the number of $\{H_1, H_2\}$ oracle queries issued by an adversary, t_e is the cost of an exponentiation in $\mathbb{G}_1$ or $\mathbb{G}_2$, and t_p is the cost of a pairing computation.

Proof Suppose that there exists an adversary $\mathcal{A}$ which can break the CCA security of our IBE system. We can then build an algorithm $\mathcal{B}$ which uses $\mathcal{A}$ to solve an asymmetric DBDH problem in $\mathbb{G}$. On input $(g_1, g_1^b, g_1^c, g_2, g_2^a, g_2^b, g_2^c, T) \in \mathbb{G}_1^3 \times \mathbb{G}_2^4 \times \mathbb{G}_T$, $\mathcal{B}$ attempts to output 1 if $T = e(g_1, g_2)^{abc}$ and 0 otherwise. $\mathcal{B}$ interacts with $\mathcal{A}$ as follows.
Setup $\mathcal{B}$ selects a random element $\delta \in \mathbb{Z}_p$ and sets $u = g_2^{-a} g_2^{\delta}$ and $A = e(g_1^b, g_2^a)$. Note that $\alpha = ab \in \mathbb{Z}_p$, which is unknown to $\mathcal{B}$. Then, $\mathcal{A}$ is given the public key $\mathsf{PK} = (g_1, u, A, H_1, H_2)$, where H_1 and H_2 are modeled as random oracles.
Query Phase 1 $\mathcal{A}$ issues H_1, H_2, private key, and decryption queries. $\mathcal{B}$ responds as follows:

H_1 queries To respond to H_1 queries, $\mathcal{B}$ maintains a list of tuples $< \mathsf{ID}_i, \gamma_i, \pi_i, H_1(\mathsf{ID}_i) >$ as explained below. We refer to this list as the H_1^{list}. When $\mathcal{B}$ is given an identity ID_i as an input to H_1, $\mathcal{B}$ first scans through the H_1^{list} to see if the input ID_i appears in a tuple $< \mathsf{ID}_i, \gamma_i, \pi_i, H_1(\mathsf{ID}_i) >$. If it does, $\mathcal{B}$ responds with $H_1(\mathsf{ID}_i)$. Otherwise, $\mathcal{B}$ picks two random exponents $\gamma_i, \pi_i \in \mathbb{Z}_p$ and sets $H_1(\mathsf{ID}_i) = (g_2^a)^{\gamma_i} g_2^{\pi_i} \in \mathbb{G}_2$. $\mathcal{B}$ adds the new tuple $< \mathsf{ID}_i, \gamma_i, \pi_i, H_1(\mathsf{ID}_i) >$ to the H_1^{list} and responds with $H_1(\mathsf{ID}_i)$. Recall that the values $\{\gamma_i\}$ are information-theoretically hidden to $\mathcal{A}$'s view.

H_2 queries To respond to H_2 queries, $\mathcal{B}$ maintains a list of tuples $< W_i, Q_i, \mu_i >$ as explained below. We refer to this list as the H_2^{list}. When $\mathcal{B}$ is given values (W_i, Q_i), which is in either $\mathbb{G}_2 \times \mathbb{G}_1$ or $\mathbb{G}_T \times \mathbb{G}_1$, as an input to H_2, $\mathcal{B}$ first scans through the H_2^{list} to see if the input (W_i, Q_i) appears in a tuple $< W_i, Q_i, \mu_i >$. If it does, $\mathcal{B}$ responds with $H_2(W_i, Q_i) = \mu_i$. Otherwise, $\mathcal{B}$ picks a random exponent $\mu_i \in \mathbb{Z}_p$ and sets $H_2(W_i, Q_i) = \mu_i$. $\mathcal{B}$ adds the new tuple $< W_i, Q_i, \mu_i >$ to the H_2^{list} and responds with $H_2(W_i, Q_i)$.

Key queries When $\mathcal{B}$ is given an identity $\mathsf{ID}_i \in \mathcal{ID}$ as an input to a private key query, $\mathcal{B}$ selects a random exponent $r \in \mathbb{Z}_p$ and (implicitly) sets $\widetilde{r} = b + r \in \mathbb{Z}_p$. $\mathcal{B}$ generates key elements $(d_{0,i}, d_{1,i})$ as $d_{0,i} = (g_2^a)^{-r}(g_2^b)^{\delta} g_2^{\delta r}$ and $d_{1,i} = g_1^b g_1^r$. The validity of those elements can be verified as follows:

$$
\begin{aligned}
d_{0,i} &= (g_2^a)^{-r}(g_2^b)^{\delta} g_2^{\delta r} = g_2^{ab}(g_2^{-a} g_2^{\delta})^{b+r} = g_2^{\alpha} u^{\widetilde{r}}, \\
d_{1,i} &= g_1^b g_1^r = g_1^{\widetilde{r}}.
\end{aligned}
$$

Next, $\mathcal{B}$ refers to the H_1^{list} to find out the tuple $< \mathsf{ID}_i, \gamma_i, \pi_i, H_1(\mathsf{ID}_i) >$. At this moment, $\mathcal{B}$'s goal is to set $H_2(d_{0,i}, d_{1,i}) = \gamma_i$. Thus, if there is a tuple $< d_{0,i}, d_{1,i}, \gamma_i >$ in the H_2^{list}, $\mathcal{B}$ can continue the key query process.

In fact, $\mathcal{B}$ can make such a (favorable) tuple always exist in the H_2^{list} as follows: whenever $\mathcal{B}$ adds a new tuple $< \mathsf{ID}_i, \gamma_i, \pi_i, H_1(\mathsf{ID}_i) >$ to the H_1^{list}, $\mathcal{B}$ generates $\mathsf{sk}_{\mathsf{ID}_i}$ by choosing a random r, constructing key elements $(d_{0,i}, d_{1,i})$ as above, setting $H_2(d_{0,i}, d_{1,i}) = \gamma_i$, and adding the tuple $< d_{0,i}, d_{1,i}, \gamma_i >$ to the H_2^{list}. On the other hand, if $H_2(d_{0,i}, d_{1,i})$ has already be set to μ_i, then $\mathcal{B}$ simply adds a new tuple $< \mathsf{ID}_i, \mu_i, \pi_i, H_1(\mathsf{ID}_i) >$ to the H_1^{list}.

Without loss of generality, let the tuple $H_2(d_{0,i}, d_{1,i}) = \gamma_i$ (where γ_i is from the tuple in the H_1^{list} above) be in the H_2^{list}. $\mathcal{B}$ generates the element $d_{2,i}$ as $d_{2,i} = (g_2^b)^{\pi_i + \gamma_i \delta} g_2^{(\pi_i + \gamma_i \delta) r}$. The validity of $d_{2,i}$ can be verified as follows:

$$
\begin{aligned}
d_{2,i} &= (g_2^b)^{\pi_i + \gamma_i \delta} g_2^{(\pi_i + \gamma_i \delta) r} = \left((g_2^a)^{\gamma_i} g_2^{\pi_i} \cdot (g_2^{-a+\delta})^{\gamma_i}\right)^{b+r} \\
&= \left(H_1(\mathsf{ID}_i) u^{H_2(d_{0,i}, d_{1,i})}\right)^{\widetilde{r}}.
\end{aligned}
$$

Then, $\mathcal{B}$ responds with a private key $\mathsf{sk}_{\mathsf{ID}_i} = (d_{0,i}, d_{1,i}, d_{2,i}, \mathrm{tag}_k = \gamma_i)$ for the requested identity ID_i. Since the key generation algorithm is stateful, $\mathcal{B}$ keeps the random exponent r used for generating $\mathsf{sk}_{\mathsf{ID}_i}$.

Decryption queries When $\mathcal{B}$ is given a ciphertext $\mathsf{CT}_i = (C_{0,i}, C_{1,i}, C_{2,i})$ (as well as an identity ID_i) as an input to a decryption query, $\mathcal{B}$ first refers to the H_1^{list} to find out the tuple $< \mathsf{ID}_i, \gamma_i, \pi_i, H_1(\mathsf{ID}_i) >$. (If no tuple exists, $\mathcal{B}$ can run the H_1-query process in advance as explained above.) Next, $\mathcal{B}$ generates a private key $\mathsf{sk}_{\mathsf{ID}_i} = (d_{0,i}, d_{1,i}, d_{2,i}, \mathrm{tag}_k)$ for the identity ID_i or uses the private key $\mathsf{sk}_{\mathsf{ID}_i}$ that was generated before. Recall that $H_2(d_{0,i}, d_{1,i}) = \gamma_i = \mathrm{tag}_k$.

To check whether $H_2(C_{0,i}, C_{1,i}) = \mathrm{tag}_c \stackrel{?}{=} \mathrm{tag}_k = H_2(d_{0,i}, d_{1,i})$, $\mathcal{B}$ refers to the H_2^{list} to search for a tuple $< C_{0,i}, C_{1,i}, \widetilde{\mu}_i >$. If such a tuple regarding $(C_{0,i}, C_{1,i})$ does not exist, $\mathcal{B}$ sets $H_2(C_{0,i}, C_{1,i}) = \widetilde{\mu}_i$ by choosing a random $\widetilde{\mu}_i \in \mathbb{Z}_p$ and adds the new tuple $< C_{0,i}, C_{1,i}, \widetilde{\mu}_i >$ to the H_2^{list}.

[Case 1] If $\widetilde{\mu}_i = \gamma_i$, $\mathcal{B}$ aborts. (We refer to this event as abort1.) Notice that γ_i is from the tuple $< \mathsf{ID}_i, \gamma_i, \pi_i, H_1(\mathsf{ID}_i) >$ in the H_1^{list}. In the real decryption, the equality means that $\text{tag}_c = \text{tag}_k$ and thus the normal decryption is expected to output $\perp$, but $\mathcal{B}$ simply aborts in our simulation. We will give the reason below. Fortunately, the probability that the event abort1 happens is negligible while H_2 acts like a random oracle.

[Case 2] If $\widetilde{\mu}_i \neq \gamma_i$, $\mathcal{B}$ performs the normal decryption using $\mathsf{sk}_{\mathsf{ID}_i}$ and replies with the resulting message.

Challenge $\mathcal{A}$ outputs two messages $M_0, M_1 \in \mathbb{G}_T$ and an identity ID^* on which it wishes to be challenged. If necessary, $\mathcal{B}$ runs the algorithm for responding to H_1 query on ID^*. Let $< \mathsf{ID}^*, \gamma^*, \pi^*, H_1(\mathsf{ID}^*) >$ be the tuple in the H_1^{list} regarding the challenged identity ID^*. Notice that $\mathcal{A}$ cannot query a private key for ID^*. This means that the exponent γ^* in the tuple is not revealed to $\mathcal{A}$ (with overwhelming probability) until the Challenge phase.

$\mathcal{B}$ picks a random bit $\sigma \in \{0, 1\}$ and sets $C_0^* = M_\sigma T$ and $C_1^* = g_1^c$. It sets $H_2(C_0^*, C_1^*) = \gamma^*$. If the tuple $< C_0^*, C_1^*, \gamma_j >$ are already in the H_2^{list} and $\gamma_j \neq \gamma^*$, then $\mathcal{B}$ aborts. (We refer to this event as abort2.) Otherwise, $\mathcal{B}$ generates the ciphertext $\mathsf{CT}^* = (C_0^*, C_1^*, C_2^*) = \left(M_\sigma T, g_1^c, (g_2^c)^{\pi^* + \delta\gamma^*}\right)$. $\mathcal{B}$ (implicitly) sets $s = c$. The validity of C_2^* can then be verified as follows:

$$\begin{aligned}(g_2^c)^{\pi^*+\delta\gamma^*} &= \left((g_2^a)^{\gamma^*} g_2^{\pi^*} \cdot (g_2^{-a+\delta})^{\gamma^*}\right)^c \\ &= \left(H_1(\mathsf{ID}^*) u^{H_2(C_0^*, C_1^*)}\right)^s.\end{aligned}$$

Query Phase 2 $\mathcal{A}$ issues more $\{H_i\}_{i=1,2}$, private key, and decryption queries. $\mathcal{B}$ responds as in Query Phase 1. At this phase, however, there are challenging decryption queries $\mathcal{B}$ has to deals with. That happens when $\mathcal{A}$ issues *valid* ciphertexts such as $\mathsf{CT}_i = (C_{0,i}, C_1^*, C_{2,i})$ on either ID or ID^*, where C_1^* is the same as in CT^*. We call a ciphertext $\mathsf{CT} = (C_0, C_1, C_2)$ under an identity ID *valid* if the pairing test upon decryption holds, i.e., $e\left(C_1, H_1(\mathsf{ID}) u^{H_2(C_0, C_1)}\right) = e(g_1, C_2)$. Notice that when a queried ciphertext is $\mathsf{CT}_i = (C_{0,i}, C_{1,i}, C_{2,i})$ on either ID or ID^* where $C_{1,i} \neq C_1^*$, $\mathcal{B}$ can simply respond as in Query Phase 1. Regarding the challenging decryption queries, there are four possible cases:

[Case 1] $\mathsf{CT}_i = (C_0^*, C_1^*, C_{2,i})$ on ID^*, where $C_{2,i} \neq C_2^*$. As the ciphertext is valid, it passes the pairing test upon decryption. Thus, $\mathcal{B}$ has that $e\left(C_1^*, H_1(\mathsf{ID}^*) u^{H_2(C_0^*, C_1^*)}\right) = e(g_1, C_{2,i})$. Since $C_1^* = g_1^c$, the equation shows $C_{2,i} = (H_1(\mathsf{ID}^*) u^{H_2(C_0^*, C_1^*)})^c$, in which case $C_{2,i} = C_2^*$. This means that such a valid ciphertext in the form of $(C_0^*, C_1^*, C_{2,i})$ such that $C_{2,i} \neq C_2^*$ is not possible.

[Case 2] $\mathsf{CT}_i = (C_{0,i}, C_1^*, C_2^*)$ on ID^*, where $C_{0,i} \neq C_0^*$. This case can happen only if $\mathcal{B}$ sets $H_2(C_{0,i}, C_1^*) = \gamma^* \in \mathbb{Z}_p$. In this case, $\mathcal{B}$ aborts. (We refer to this event as abort3.) This is because otherwise, i.e., $\mathcal{B}$ returns $\perp$ as the normal decryption result and this gives the information of γ^* (as tag_k) in the $\mathsf{sk}_{\mathsf{ID}^*}$. γ^* (as tag_c^*) is already used for the challenge ciphertext, which gives the knowledge that γ^* is used two times in both $\mathsf{sk}_{\mathsf{ID}^*}$ and CT^*. Naturally, such an unusual leakage can cause $\mathcal{A}$ to distinguish between the simulation and the real attack.

[Case 3] $\mathsf{CT}_i = (C_{0,i}, C_1^*, C_{2,i})$ on ID^*, where $C_{0,i} \neq C_0^*$ and $C_{2,i} \neq C_2^*$. As the ciphertext is valid, $\mathcal{B}$ has that $e\left(C_1^*, H_1(\mathsf{ID}^*) u^{H_2(C_{0,i}, C_1^*)}\right) = e(g_1, C_{2,i})$. Since $C_1^* = g_1^c$, the equation shows that $C_{2,i} = (H_1(\mathsf{ID}^*) u^{H_2(C_{0,i}, C_1^*)})^c$. Also, since $C_{2,i} \neq C_2^*$, we know that $H_2(C_{0,i}, C_1^*) \neq \gamma^*$. Then, $\mathcal{B}$ has that

$$\begin{aligned}C_{2,i} &= \left(H_1(\mathsf{ID}^*) u^{H_2(C_{0,i}, C_1^*)}\right)^s \\ &= \left((g_2^a)^{\gamma^*} g_2^{\pi^*} \cdot (g_2^{-a+\delta})^{H_2(C_{0,i}, C_1^*)}\right)^c\end{aligned}$$

$$= (g_2^{ac})^{\gamma^* - H_2(C_{0,i}, C_1^*)} (g_2^c)^{\pi^* + \delta H_2(C_{0,i}, C_1^*)},$$

in which case $\mathcal{B}$ can obtain g_2^{ac} by computing $\left[C_{2,i}/(C_1^*)^{\pi^* + \delta H_2(C_{0,i}, C_1^*)}\right]^{1/(\gamma^* - H_2(C_{0,i}, C_1^*))}$. It follows that $\mathcal{B}$ can solve the asymmetric DBDH problem immediately.

[Case 4] $\mathsf{CT}_i = (C_{0,i}, C_1^*, C_{2,i})$ on $\mathsf{ID} (\neq \mathsf{ID}^*)$. Let $< \mathsf{ID}, \gamma, \pi, H_1(\mathsf{ID}) >$ be the tuple in the H_1^{list} regarding ID. From the pairing test equation, $\mathcal{B}$ has that $e\left(C_1^*, H_1(\mathsf{ID}) u^{H_2(C_{0,i}, C_1^*)}\right) = e(g_1, C_{2,i})$. Since $C_1^* = g_1^c$, the equation shows that $C_{2,i} = (H_1(\mathsf{ID}) u^{H_2(C_{0,i}, C_1^*)})^c$. If $H_2(C_{0,i}, C_1^*) = \gamma$, $\mathcal{B}$ outputs $\perp$ as the result of normal decryption.[2] Otherwise, $\mathcal{B}$ has that

$$\begin{aligned} C_{2,i} &= \left(H_1(\mathsf{ID}) u^{H_2(C_{0,i}, C_1^*)}\right)^s \\ &= \left((g_2^a)^\gamma g_2^\pi \cdot (g_2^{-a+\delta})^{H_2(C_{0,i}, C_1^*)}\right)^c \\ &= (g_2^{ac})^{\gamma - H_2(C_{0,i}, C_1^*)} (g_2^c)^{\pi + \delta H_2(C_{0,i}, C_1^*)}, \end{aligned}$$

in which case $\mathcal{B}$ can obtain g_2^{ac} by computing $\left[C_{2,i}/(C_1^*)^{\pi + \delta H_2(C_{0,i}, C_1^*)}\right]^{1/(\gamma - H_2(C_{0,i}, C_1^*))}$. It follows that $\mathcal{B}$ can solve the asymmetric DBDH problem immediately.

Guess $\mathcal{A}$ outputs a guess $\sigma' \in \{0, 1\}$. $\mathcal{B}$ then outputs its guess $\sigma' \in \{0, 1\}$ as the solution to the asymmetric DBDH problem.

Comment The reason why $\mathcal{B}$ aborts in the event $\mathsf{abort1}$ is that the equality can leak the information on the tag_k such that $H_2(d_{0,i}, d_{1,i}) = \gamma_i = \mathrm{tag}_k$, where γ_i is from the tuple $< \mathsf{ID}_i, \gamma_i, \pi_i, H_1(\mathsf{ID}_i) >$ and $(d_{0,i}, d_{1,i})$ are from the private key $\mathsf{sk}_{\mathsf{ID}_i} = (d_{0,i}, d_{1,i}, d_{2,i}, \mathrm{tag}_k = \gamma_i)$. That is, $\mathcal{A}$ is able to know that the value $\mathrm{tag}_k (= \gamma_i)$ is used in $\mathsf{sk}_{\mathsf{ID}_i}$, even though all key elements in $\mathsf{sk}_{\mathsf{ID}_i}$ are not revealed to $\mathcal{A}$. The problem can then happen in the Challenge phase where $\mathcal{A}$ can select such ID_i as the challenge identity. Then, $\mathcal{B}$ has no choice but to use the same γ_i as the tag_c^* in constructing the challenge ciphertext such that $H_2(C_0^*, C_1^*) = \gamma_i$. This gives the information that γ_i is used two times in both $\mathsf{sk}_{\mathsf{ID}_i}$ and CT^*. As mentioned above, such an unusual leakage can cause $\mathcal{A}$ to distinguish between the simulation and the real attack.

Analysis The dominated additional computation that $\mathcal{B}$ requires is both the exponentiations for handling q_K private key queries and the pairings for handling q_D decryption queries. Thus, the inequality about computational time can easily be obtained.

Next, we assume Cases 3 and 4 described in the Query Phase 2 do not happen (which would rather increase $\mathcal{B}$'s success probability to solve the asymmetric DBDH problem). To analyze $\mathcal{B}$'s advantage, we first prove the following claim that argues that the probability that $\mathcal{B}$ aborts in the simulation is at most $\frac{q_{H_2}}{p} + \frac{2q_D}{p}$, which is negligible. □

Claim 1 $\Pr[\mathsf{abort}] = \Pr[\mathsf{abort1} \vee \mathsf{abort2} \vee \mathsf{abort3}]$ *in the simulation is at most* $\frac{q_{H_2}}{p} + \frac{2q_D}{p}$.

Proof The event $\mathsf{abort1}$ happens if $\mathcal{B}$ sets $H_2(C_{0,i}, C_{1,i}) = \gamma_i$ for any queried ciphertext $\mathsf{CT}_i = (C_{0,i}, C_{1,i}, C_{2,i})$, where γ_i is from the tuple $< \mathsf{ID}_i, \gamma_i, \pi_i, H_1(\mathsf{ID}_i) >$ in the H_1^{list}. γ_i is a pre-determined value and the output of H_2 query is just set by choosing a random value in $\mathbb{Z}_p$. Thus, the probability that the event $\mathsf{abort1}$ happens is at most $1/p$ on each decryption query. Since $\mathcal{B}$ has to handle q_D decryption queries, the probability that the event $\mathsf{abort1}$ occurs throughout the simulation becomes at most q_D/p.

The event $\mathsf{abort2}$ can occur if the value $(M_\sigma T, g^c)$ already exists in the H_2^{list} as the input value queried by $\mathcal{A}$. There are p possibilities in picking a value as an input to the H_2 query, because it is determined by a randomly chosen exponent $c \in \mathbb{Z}_p$. This gives the probability

[2] After the Challenge phase, we do not need to consider the event $\mathsf{abort1}$ in the case of $\mathsf{ID} (\neq \mathsf{ID}^*)$, since the distribution of tag values regarding $\mathsf{ID} (\neq \mathsf{ID}^*)$ is statistically identical to that in the real attack.

at most q_{H_2}/p that the event $\mathsf{abort2}$ happens. (If we consider the possible cases from the selection of a value in $\mathbb{G}_T$, then the probability will be much smaller.)

Regarding the event $\mathsf{abort3}$, the event happens if $\mathcal{B}$ sets $H_2(C_{0,i}, C_1^*) = \gamma^*$ for any queried ciphertext $\mathsf{CT}_i = (C_{0,i}, C_1^*, C_2^*)$ on ID^*, where $C_{0,i} \neq C_0^*$. Here, the value γ^* is the pre-determined value and the output of H_2 query is just set by choosing a random value in $\mathbb{Z}_p$. Thus, the probability that the event $\mathsf{abort3}$ happens is at most $1/p$ on each decryption query. Since $\mathcal{B}$ has to handle q_D decryption queries, the probability that the event $\mathsf{abort3}$ occurs throughout the simulation becomes at most q_D/p.

We know that all the events that $\mathcal{B}$ aborts are relatively independent. As a result, the probability $\Pr[\mathsf{abort1} \vee \mathsf{abort2} \vee \mathsf{abort3}]$ in the simulation is at most $\frac{q_{H_2}}{p} + \frac{2q_D}{p}$.

From Claim 1, we can see that the probability that $\mathcal{B}$ aborts in the simulation is negligible (under the appropriate selection of security parameters). We argue that as long as $\mathcal{B}$ does not abort, $\mathcal{B}$ provides $\mathcal{A}$ with a perfect simulation whose distribution is identical to the distribution in a real attack. This is because (1) the simulation of H_1 and H_2 oracles are obviously perfect as the output values are determined by randomly chosen values in $\mathbb{G}_2$ and $\mathbb{Z}_p$, respectively, and (2) the simulation of private key oracles is also perfect as each key on an identity ID_i is generated with a randomly chosen exponent $r \in \mathbb{Z}_p$ such that $\widetilde{r} = b + r$, and (3) the simulation of decryption oracles is also perfect as it is done via normal decryption using private keys, and (4) the values $\{\gamma_i\}$ in the H_1^{list} are uniformly distributed and information-theoretically hidden from $\mathcal{A}$'s view until private keys and CT^* are given to $\mathcal{A}$.

As long as $\mathcal{B}$ does not abort in the simulation, $\mathcal{B}$ can use the $\mathcal{A}$'s advantage to break the chosen ciphertext security straightforwardly. This can be checked as follows: if $T = e(g_1, g_2)^{abc}$, then the challenge ciphertext CT^* is a valid encryption of M_σ under ID^*. Otherwise, i.e., if T is random in $\mathbb{G}_T$, then $M_\sigma T$ is independent of the bit σ. Thus, if $\mathcal{A}$ distinguishes between the two ciphertexts, then $\mathcal{B}$ can distinguish between the two possible values of T with the same advantage. Therefore, unless $\mathcal{B}$ does not abort, we have the following result:

$$\mathbf{Adv}^{\mathrm{CCA}}_{\mathcal{IBE},\mathcal{A}}(k) \leq \left(1 - \frac{q_{H_2}}{p} - \frac{2q_D}{p}\right) \cdot \mathbf{Adv}^{\mathrm{ADBDH}}_{\mathbb{G},\mathcal{B}}(k),$$

as required. This concludes the proof of Theorem 1. □

3.3 Hash-BDH and BDH construction

Our IBE system can be slightly modified to encrypt arbitrary n-bit message strings such as $M \in \{0,1\}^n$. To do this, we consider a family of hash functions of the form $H_3 : \mathbb{G}_T \to \{0,1\}^n$ (where $n \in \mathbb{Z}^+$ is determined by the security parameter). A resultant ciphertext is then computed as $C_0 = H_3(A^s) \oplus M \in \{0,1\}^n$, $C_1 = g_1^s \in \mathbb{G}_1$, and $C_3 = \left(H_1(\mathsf{ID})u^{\mathrm{tag}_c}\right)^s \in \mathbb{G}_2$ where $\mathrm{tag}_c = H_2(C_0, C_1) \in \mathbb{Z}_p$. Decryption can be performed by hashing the pairing value in the equation (2) and XOR-ing the result with C_0.

The security of the modified system can be proven in two flavors: if H_3 is a random selection of the (appropriate) hash family, then the modified system is IND-ID-CCA secure under the asymmetric Hash-BDH assumption[3] and the security reduction becomes tight. The proof is almost identical to that of Theorem 1. On the other hand, if H_3 is modeled as a random oracle, then the modified system is IND-ID-CCA secure under the asymmetric BDH assumption[4] and the security loss becomes $O(q_{H_3})$. In this case, $\mathcal{B}$ maintains additional

[3] This is the asymmetric version of the Hash-BDH assumption [10].

[4] This is the asymmetric version of the BDH assumption [13].

H_3^{list} with respect to H_3, and the challenge ciphertext is constructed as $\mathsf{CT}^* = (C_0^* = R, C_1^* = g_1^c, C_2^* = \left(H_1(\mathsf{ID}^*)u^{\mathrm{tag}_c^*}\right)^c)$ where R is a randomly chosen string in $\{0,1\}^n$ and $\mathrm{tag}_c^* = H_2(C_0^*, C_1^*)$. C_0^* is not relevant to any of two challenged messages, and $\mathcal{B}$ just wants to employ the adversary's advantage to issue the correct answer of an asymmetric BDH problem as the input of H_3 query. At the end of the simulation, $\mathcal{B}$ selects a random input value among tuples in the H_3^{list} as its answer to the BDH problem, which causes the security loss of $O(q_{H_3})$. We notice that CCA security using a similar proof strategy was already used to prove IND-ID-CPA security of 'BasicIdent' in [13]. The rest of the proof can be completed based on the proof of the BasicIdent as well as Theorem 1.

4 Extension for arbitrary length messages

We extend our IBE system to deal with arbitrary length messages. Our extended system is based on the well-known framework using the key encapsulation mechanism (KEM) and data encapsulation mechanism (DEM). Identity-based KEM (IBKEM) encrypts a symmetric key under which an arbitrarily long message is encrypted under a symmetric-key cipher DEM. Usually, to achieve CCA security of an entire IBE system, both IBKEM and DEM should be CCA-secure respectively [8] or DEM should be an authenticated symmetric-key encryption [39]. However, a slight difference resides in our extended IBE system where it is sufficient for DEM to be a one-time symmetric-key encryption secure against passive attacks [25]. In practice, such a weak DEM can easily be instantiated with a block cipher using a so-called 'counter mode'. The difference is because our IBE system is able to provide a consistency check (using pairing) to see if ciphertext elements including the DEM part are the same as what an encryptor constructed. Such a weak DEM when achieving CCA security can also be seen in the CCA-secure BF-IBE system [13].

4.1 Construction

Setup(k)**:** As in the previous IBE system. Additionally, the setup algorithm chooses a one-time symmetric-key encryption scheme $\mathcal{SKE} = (\mathcal{E}, \mathcal{D})$. The public parameters PP and the master secret key msk are generated as

$$\mathsf{PP} = \left(g_1, u, A, H_1, H_2, \mathcal{SKE}\right), \quad \mathsf{msk} = g_2^{\alpha}.$$

KeyGen$(\mathsf{msk}, \mathsf{PP}, \mathsf{ID})$**:** As in the previous IBE system.
Encrypt$(\mathsf{PP}, \mathsf{ID}, M)$**:** To encrypt an arbitrary length message $M \in \{0,1\}^*$ under an identity $\mathsf{ID} \in \mathcal{ID}$, the encryption algorithm does as follows:

1. Pick a random exponent $s \in \mathbb{Z}_p$.
2. Compute a key $K = A^s \in \mathbb{G}_T$.
3. Compute $C_0 = \mathcal{E}_K(M)$, $C_1 = g_1^s \in \mathbb{G}_1$, and $\mathrm{tag}_c = H_2(C_0, C_1) \in \mathbb{Z}_p$.
4. Compute $C_2 = \left(H_1(\mathsf{ID})u^{\mathrm{tag}_c}\right)^s \in \mathbb{G}_2$.
5. Output a ciphertext $\mathsf{CT} = (C_0, C_1, C_2) \in \{0,1\}^{|M|} \times \mathbb{G}_1 \times \mathbb{G}_2$.

Decrypt$(\mathsf{PP}, \mathsf{CT}, \mathsf{sk}_{\mathsf{ID}})$**:** To decrypt a ciphertext (C_0, C_1, C_2) using a private key $\mathsf{sk}_{\mathsf{ID}} = (d_0, d_1, d_2, \mathrm{tag}_k)$ for ID, the decryption algorithm does as follows:

1. Compute $\mathrm{tag}_c = H_2(C_0, C_1)$ and check whether or not $e\left(C_1, H_1(\mathsf{ID})u^{\mathrm{tag}_c}\right) \stackrel{?}{=} e(g_1, C_2)$.
2. If the equality above fails, choose a random key $\widetilde{K}$ from $\mathbb{G}_T$ and output $\mathcal{D}_{\widetilde{K}}(C_0)$.
3. Otherwise, check if $\mathrm{tag}_c \stackrel{?}{=} \mathrm{tag}_k$.

4. If the equality above holds, output $\perp$.
5. Otherwise, compute a key

$$K = e(C_1, d_0) \cdot \left(\frac{e(d_1, C_2)}{e(C_1, d_2)}\right)^{\frac{-1}{\mathrm{tag}_C - \mathrm{tag}_k}}.$$

6. Output a message $M = \mathcal{D}_K(C_0)$.

Remark The efficiency of the IBE system above is almost the same as that in the previous section. Notice that the KEM part in a ciphertext is not expanded and the DEM part $C_0 = \mathcal{E}_K(M)$ is also hashed and embedded into the ciphertext element C_3. We can check the consistency of ciphertext elements including the DEM part and therefore avoid to use an authenticated encryption scheme with the help of a secure message authentication code (MAC). In practice, a one-time symmetric-key encryption scheme $\mathcal{SKE}$ with key-space $\mathcal{K} \in \{0,1\}^k$ can be implemented by AES with a counter mode, and a real symmetric key for $\mathcal{E}$ can be obtained via a key-derivation function [25] that maps an element from $\mathbb{G}_T$ into $2k$-bit strings.

4.2 Security

Theorem 2 *Let H_1 and H_2 be modeled as random oracles. Suppose the (t', ϵ')-asymmetric DBDH assumption holds in $\mathbb{G}_1$ and $\mathbb{G}_2$ and the one-time symmetric-key encryption scheme $\mathcal{SKE}$ is (t'', ϵ'')-secure against passive attacks. Then our IBE system is (t, ϵ, q_K, q_D)-IND-ID-CCA secure, where*

$$\epsilon \leq \left(1 - \frac{q_{H_2}}{p} - \frac{2q_D}{p}\right) \cdot (\epsilon' + \epsilon''),$$
$$t \geq t' + t'' - \mathcal{O}((q_K + q_D + q_{H_1})t_e) - \mathcal{O}(q_D(t_p + t_{symD})).$$

Here, $\{q_{H_1}, q_{H_2}\}$ is the number of $\{H_1, H_2\}$ oracle queries issued by an adversary, t_e is the cost of an exponentiation in $\mathbb{G}_1$ or $\mathbb{G}_2$, t_p is the cost of a pairing computation, and t_{symD} is the cost of symmetric-key decryption in the $\mathcal{SKE}$.

Proof The proof of Theorem 2 is almost similar to that of Theorem 1. For clarity, we reconstruct the entire proof by creating a sequence of hybrid games. If necessary, we will adapt several notations that appeared in the proof of Theorem 1 without explicit explanation.

Let $\mathcal{A}$ be an adversary on the chosen ciphertext security of our IBE system above. We will consider two games, **Game 0** and **Game 1**, each game interacting with $\mathcal{A}$. Let X_i (for $i = 0, 1$) be the event that in **Game i**, $\mathcal{A}$ wins the game.

Game 0 The game is a real attack game of chosen ciphertext security, so that (for a security parameter k) we have

$$|\Pr[X_0] - 1/2| = \mathbf{Adv}^{\mathrm{CCA}}_{\mathcal{IBE},\mathcal{A}}(k). \tag{3}$$

Game 1 The game is the same as Game 0, except that the value K^* used for a symmetric key in CT^* is replaced by a random value $K \in \mathbb{G}_T$. Unless the events abort1 and abort2 and abort3 (defined in the proof of Theorem 1) occur in **Game 1**, we have

$$|\Pr[X_1] - \Pr[X_0]| = \mathbf{Adv}^{\mathrm{ADBDH}}_{\mathbb{G},\mathcal{B}_1}(k), \tag{4}$$

where $\mathcal{B}_1$ is an algorithm whose goal is to solve the asymmetric DBDH problem.

The proof of (4) is almost the same as that of Theorem 1. On an input $(g_1, g_1^b, g_1^c, g_2, g_2^a, g_2^b, g_2^c, T)$, we describe how $\mathcal{B}_1$ constructs CT^*: when $\mathcal{A}$ outputs two messages $M_0, M_1 \in$

$\{0,1\}^*$ of the same length and an identity ID^*, $\mathcal{B}_1$ picks a random bit $\sigma \in \{0,1\}$. Let $< \mathsf{ID}^*, \gamma^*, \pi^*, H_1(\mathsf{ID}^*) >$ be the tuple in the H_1^{list} regarding ID^*. $\mathcal{B}_1$ sets $C_0^* = \mathcal{E}_T(M_\sigma)$, $C_1^* = g_1^c$, and $H_2(C_0^*, C_1^*) = \gamma^*$. If the tuple $< C_0^*, C_1^*, \gamma_j >$ are already in the H_2^{list} and $\gamma_j \neq \gamma^*$, then $\mathcal{B}_1$ aborts. (In the proof of Theorem 1 (and thus Game 1 above), the event has already been taken into consideration under the event $\mathsf{abort2}$.) Otherwise, the challenge ciphertext is generated as $\mathsf{CT}^* = (C_0^*, C_1^*, C_2^*) = \left(\mathcal{E}_T(M_\sigma), g_1^c, (g_2^c)^{\pi^* + \delta\gamma^*}\right)$. Other slight differences exist in handling decryption and H_2 queries. Answering decryption queries needs to run $\mathcal{D}$ of $\mathcal{SKE}$, and inputs (W_i, Q_i) to H_2 queries are in either $\mathbb{G}_2 \times \mathbb{G}_1$ or $\{0,1\}^* \times \mathbb{G}_1$ (instead of $\mathbb{G}_2 \times \mathbb{G}_1$ or $\mathbb{G}_T \times \mathbb{G}_1$). However, the way of answering H_2 queries is the same as in the proof of Theorem 1. Thus, the probability that the events $\mathsf{abort1}$ and $\mathsf{abort2}$ and $\mathsf{abort3}$ happen in proving the Eq. 4 is the same as that in Theorem 1.

Finally, in **Game 1** we have the following relation that directly comes from the definition of ciphertext indistinguishability for $\mathcal{SKE}$ unless the event $\mathsf{abort3}$ occurs in **Game 1**:

$$|\Pr[X_1] - 1/2| = \mathbf{Adv}^{\text{OT-IND}}_{\mathcal{SKE},\mathcal{B}_2}(k), \tag{5}$$

where $\mathcal{B}_2$ is an algorithm whose goal is to break $\mathcal{SKE}$.

One thing we want to clarify is that $\mathcal{B}_2$ does not need to deal with any decryption query regarding the one-time symmetric-key encryption scheme $\mathcal{SKE}$. To prove the Eq. 5, $\mathcal{B}_2$ relays the two messages $M_0, M_1 \in \{0,1\}^*$ to its challenger and is given a challenge ciphertext $\mathcal{E}_{K^*}(M_\sigma)$ for a random (unknown) key $K^* \in \mathbb{G}_T$. $\mathcal{B}_2$ then reconstructs its challenge ciphertext CT^* as explained above and gives it to $\mathcal{A}$. Whenever $\mathcal{A}$ requests any decryption query on CT_i, $\mathcal{B}_2$ can use private keys (generated by $\mathcal{B}_2$) to perform normal decryption. The only troublesome queries are possible when $\mathcal{A}$ issues decryption queries on valid $\mathsf{CT}_i = (C_{0,i}, C_1^*, C_{2,i})$, in which case $\mathcal{B}_2$ would have to decrypt CT_i with the *unknown* symmetric-key K^*. Fortunately, those troublesome queries can be handled as in Query Phase 2 of Theorem 1. More precisely, there are four possible cases:

[Case 1] $\mathsf{CT}_i = (C_0^*, C_1^*, C_{2,i})$ on ID^*, where $C_{2,i} \neq C_2^*$. As before, it is impossible to generate a valid ciphertext $(C_0^*, C_1^*, C_{2,i})$ such that $C_{2,i} \neq C_2^*$.

[Case 2] $\mathsf{CT}_i = (C_{0,i}, C_1^*, C_2^*)$ on ID^*, where $C_{0,i} \neq C_0^*$. This happens only if $\mathcal{B}$ sets $H_2(C_{0,i}, C_1^*) = \text{tag}_c^* \in \mathbb{Z}_p$. In this case, $\mathcal{B}$ aborts. (This event has already been referred to as $\mathsf{abort3}$.)[5]

[Case 3] $\mathsf{CT}_i = (C_{0,i}, C_1^*, C_{2,i})$ on ID^*, where $C_{0,i} \neq C_0^*$ and $C_{2,i} \neq C_2^*$. We have shown that the $\mathcal{A}$'s ability to issue such a valid ciphertext can be used to compute g_2^{ac} in the previous Query Phase 2.

[Case 4] $\mathsf{CT}_i = (C_{0,i}, C_1^*, C_{2,i})$ on $\mathsf{ID}(\neq \mathsf{ID}^*)$. $\mathcal{B}$ generates $\mathsf{sk}_{\mathsf{ID}}$ and performs normal decryption. If $H_2(C_{0,i}, C_1^*) = \text{tag}_k$ where tag_k is from $\mathsf{sk}_{\mathsf{ID}}$, then $\mathcal{B}$ outputs $\perp$ as the result of normal decryption. Otherwise, we also have shown that the $\mathcal{A}$'s ability to issue such a valid ciphertext can be used to compute g_2^{ac} in the previous Query Phase 2.

Analysis It is easy to see that time complexity is obviously achieved as argued in Theorem 2. As long as the events $\mathsf{abort1}$ and $\mathsf{abort2}$ and $\mathsf{abort3}$ do not happen throughout the simulation, the Eqs. 4 and 5 hold. As a result, by putting the results of all relations 3, 4, and 5 above together, we gain a bound on the $\mathcal{A}$'s advantage as follows:

$$\mathbf{Adv}^{\text{CCA}}_{\mathcal{IBE},\mathcal{A}}(k) \leq \left(1 - \frac{q_{H_2}}{p} - \frac{2q_D}{p}\right) \cdot \left(\mathbf{Adv}^{\text{ADBDH}}_{\mathbb{G},\mathcal{B}_1}(k) + \mathbf{Adv}^{\text{OT-IND}}_{\mathcal{SKE},\mathcal{B}_2}(k)\right),$$

which concludes the proof of Theorem 2. □

[5] Notice that $\mathcal{B}$ can handle this ciphertext if the one-time symmetric-key encryption scheme $\mathcal{SKE}$ is CCA-secure. However, we simply consider $\mathcal{SKE}$ as being secure against passive attacks by adding q_D/p into the probability that $\mathcal{B}_2$ aborts.

Table 1 Security comparison between the previous IBE systems and ours

	Assumption[a]	Security	Security reduction	
			Asymptotic bound	Concrete loss[b]
BF [13]	DBDH	IND-ID-CCA	$q_K q_H$	$>2^{50}$
BB_1 [10][c]	DBDH	IND-ID-CCA	q_H	2^{50}
SK [20]	q-DBDHI	IND-ID-CCA	q_H^2	$>2^{50}$
AGF^+ [5]	DBDH	IND-ID-CCA	1	1
Cor^{FO} [24]	DBDH	IND-ID-CCA	1	1
Ours	DBDH	IND-ID-CCA	1	1

$q \approx q_H + q_K$ in [20]
q_H the number of (random-oracle) hash queries, q_K the number of private key queries
[a] All the assumptions are considered in asymmetric setting
[b] Estimated with $q_H = 2^{50}$ and $q_K = 2^{25}$
[c] We borrow the CCA-secure variant of BB_1 system from [16]

5 Comparison to previous IBE systems

In this section we compare our IBE system with the previous practical IBE systems such as BF [13], SK [20,22,43], the CCA-secure variant [16][6] of BB_1, Attrapadung et al.'s IBE system [5] (denoted as 'AFG^+'), and the CCA-secure variant[7] of [24] (denoted as 'Cor^{FO}') in terms of security and efficiency. Following Bellare and Rogaway [6,39], we estimate the number of (random oracle) hash queries as $q_H = 2^{50}$ and the number of private key queries as $q_K = 2^{25}$. For a fair comparison we consider the asymmetric decisional type of security assumptions in each system, so that BF, BB_1, AFG^+, and Cor^{FO} systems are based on the asymmetric DBDH assumption and the SK system relies on the asymmetric version of q-decisional bilinear Diffie–Hellman inversion (DBDHI) assumption [10]. We also refer to [28] for correcting a flawed security analysis of BF system. Table 1 presents the comparison result with respect to security assumptions and reductions, which was also addressed by [16,24,39]. The 'asymptotic bound' in the security reduction means that the advantage of breaking the CCA security of an IBE system is larger than that of solving a security assumption in comparable time, by a factor of each bound. In other words, the larger the bound is, the bigger the security loss (i.e., gap) is between an IBE system and a security assumption. In Table 1, a concrete bound at each IBE system is estimated when considering the reasonable number of adversarial queries q_H and q_K as 2^{50} and 2^{25}, respectively. The respective bound tells us that, roughly speaking,

$$\mathbf{Adv}^{\mathrm{CCA}}_{\mathcal{IBE},\mathcal{A}} \leq (\text{bound}) \cdot \mathbf{Adv}^{\mathrm{Assumption}}_{\mathcal{B}} \tag{6}$$

for algorithms $\mathcal{A}$ and $\mathcal{B}$. When the bound is quite large, we have to choose a larger security parameter k for a security assumption so that we can make $(\text{bound}) \cdot \mathbf{Adv}^{\mathrm{Assumption}}_{\mathcal{B}}$ small and consequently $\mathbf{Adv}^{\mathrm{CCA}}_{\mathcal{IBE},\mathcal{A}}$ small enough at a desired security level. This is the reason why one has to choose a larger system security parameter than an idealized security level when a large

[6] We are not sure that the CCA security proof about the variant is correct because we cannot find any security proof for the variant. Boyen [16] stated that the proof of CCA security was adapted by [10], but in any part of [10] there exists no security proof related to the variant.

[7] The variant was proven to be chosen-plaintext secure under the DBDH assumption, but we consider the CCA-secure version by applying the Fujisaki–Okamoto transform [27].

Table 2 Efficiency comparison between the previous IBE systems and ours

	PP	CT	KGen	Enc	Dec
BF [13]	$2\mathtt{G}_1$	$1\mathtt{G}_1, 1\mathtt{h}$	$1\mathtt{H}_2, 1\mathtt{E}_2$	$1\mathtt{H}_2, 2\mathtt{E}_1^{\mathtt{f}}, 1\mathtt{P}$	$1\mathtt{E}_1^{\mathtt{f}}, 1\mathtt{P}$
BB_1 [10][a]	$3\mathtt{G}_1, 1\mathtt{G}_\mathtt{T}$	$2\mathtt{G}_1, 1\|p\|$	$2\mathtt{E}_2^{\mathtt{f}}$	$3\mathtt{E}_1^{\mathtt{f}}, 1\mathtt{E}_\mathtt{T}^{\mathtt{f}}$	$1\mathtt{E}_1^{\mathtt{f}}, 1\mathtt{E}_\mathtt{T}^{\mathtt{f}}, 2\mathtt{P}^{\sharp}$
SK [20]	$2\mathtt{G}_1, 1\mathtt{G}_\mathtt{T}$	$1\mathtt{G}_1, 1\mathtt{h}$	$1\mathtt{E}_2^{\mathtt{f}}$	$2\mathtt{E}_1^{\mathtt{f}}, 1\mathtt{E}_\mathtt{T}^{\mathtt{f}}$	$2\mathtt{E}_1^{\mathtt{f}}, 1\mathtt{P}$
AGF^+ [5]	$2\mathtt{G}_1$	$1\mathtt{G}_1, 2\mathtt{h}$	$1\mathtt{H}_2, 1\mathtt{E}_2$	$2\mathtt{H}_2, 2\mathtt{E}_1^{\mathtt{f}}, 2\mathtt{P}$	$2\mathtt{E}_1^{\mathtt{f}}, 2\mathtt{P}$
Cor^{FO} [24]	$2\mathtt{G}_1, 1\mathtt{G}_\mathtt{T}$	$1\mathtt{G}_1, 1\mathtt{G}_\mathtt{T}, 1\mathtt{h}$	$1\mathtt{H}_2, 1\mathtt{E}_2 1\mathtt{E}_2^{\mathtt{f}}$	$1\mathtt{H}_2, 2\mathtt{E}_1^{\mathtt{f}}, 1\mathtt{E}_\mathtt{T}^{\mathtt{f}}, 1\mathtt{P}$	$1\mathtt{E}_1^{\mathtt{f}}, 1\mathtt{P}, 1\mathtt{E}_\mathtt{T}^{\mathtt{f}}, 1\mathtt{E}_\mathtt{T}$
Ours	$1\mathtt{G}_1, 1\mathtt{G}_2, 1\mathtt{G}_\mathtt{T}$	$1\mathtt{G}_1, 1\mathtt{G}_2$	$1\mathtt{H}_2, 1\mathtt{E}_2, 1\mathtt{E}_1^{\mathtt{f}}, 2\mathtt{E}_2^{\mathtt{f}}$	$1\mathtt{H}_2, 1\mathtt{E}_2, 1\mathtt{E}_1^{\mathtt{f}}, 1\mathtt{E}_2^{\mathtt{f}}, 1\mathtt{E}_\mathtt{T}^{\mathtt{f}}$	$2\mathtt{E}_1^{\mathtt{f}}, 3\mathtt{E}_2^{\mathtt{f}}, 2\mathtt{P}$[b]

$\{\mathtt{G}_1, \mathtt{G}_2, \mathtt{G}_\mathtt{T}\}$ element size in $\{\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T\}$, respectively, $\mathtt{h}$ output size of hash function, p group order, $\mathtt{H}_2$ map-to-point hash into $\mathbb{G}_2$, $\{\mathtt{E}_2, \mathtt{E}_\mathtt{T}\}$ exponentiation in $\{\mathbb{G}_2, \mathbb{G}_T\}$, $\{\mathtt{E}_1^{\mathtt{f}}, \mathtt{E}_2^{\mathtt{f}}, \mathtt{E}_\mathtt{T}^{\mathtt{f}}\}$ fixed-base exponentiation in $\{\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T\}$, respectively, $\mathtt{P}$ pairing

[a] Two parings can be optimized into about 1.2 pairing

[b] We borrow the CCA-secure variant of BB_1 system from [16].

Table 3 Representation sizes and estimated calculation times for various algebraic operations

Curves / security level	Representation sizes (bits)				Relative timings[a]									
	$\mathbb{Z}_p$	$\mathbb{G}_1$	$\mathbb{G}_2$	$\mathbb{G}_T$	$\mathbb{G}_1$			$\mathbb{G}_2$			$\mathbb{G}_T$			
					$\mathtt{E}^{\mathtt{f}}$	$\mathtt{E}$	$\mathtt{H}$	$\mathtt{E}^{\mathtt{f}}$	$\mathtt{E}$	$\mathtt{H}$	$\mathtt{E}^{\mathtt{f}}$	$\mathtt{E}$	$\mathtt{P}$	$\mathtt{P}_\mathtt{r}$
SS / 80	160	512	512	1024	2	10	10	2	10	10	2	10	100	120
MNT / 80	160	171	1026	1026	0.2	1	1	8	40	40	2	10	100	120
MNT / 128	256	512	3072	3072	3	15	15	100	500	500	30	150	1500	1800

$\mathtt{E}^{\mathtt{f}}$ fix-base exponentiation, $\mathtt{E}$ general exponentiation, $\mathtt{H}$ map-to-point hashing, $\mathtt{P}$ pairing, $\mathtt{P}_\mathtt{r}$ a ratio of two pairings

[a] The time unit is defined as the cost of a general exponentiation (i.e., point multiplication) on a random MNT curve at the 80-bit security level

security loss arises at reduction. In contrast to the BF, SK, and BB_1 systems, AFG^+, Cor^{FO} and our systems have *tight* security reductions to the asymmetric DBDH assumption and thus we can say that the latter IBE systems are as secure as the hardness of the asymmetric DBDH assumption. We notice that security of all the compared IBE systems above is proven in the random oracle model.

Table 2 presents an efficiency comparison between the previous IBE systems and ours, considering the space overheads of the various data types and the number of group operations. To give more detailed comparison based on the security loss, we employ Boyen's work [16] that investigates relatively estimated calculation times for various operations and representation sizes for group elements. Table 3 shows the values when considering SS curves at the 80-bit security level and MNT curves at security levels 80 and 128. Especially, following [16], we consider that a ratio of two pairings can be optimized into about 1.2 pairing in all the three cases. We focus on the 80-bit security as a desired security level of IBE system. As mentioned above, BF, SK, and BB_1 systems all have concrete security loss of at least 2^{50}, so that they must have a larger system security parameter than 80 bits. Obviously, it will be unfair to straightforwardly compare ours with the three systems at the same security level. As a warm-up case, however, we give an efficiency comparison when considering supersin-

Table 4 Estimated calculation times for CCA-secure IBE systems at 80-bit security level (not considering security loss)

	Curves / k	Overheads (bits)		Relative timings[a]		
		PP	CT[b]	KGen	Enc	Dec
BF	SS/80	1024	672	20	114	102
BB_1	SS/80	2560	1184	4	8	204 (124[b])
SK	SS/80	2048	672	2	6	104
AFG^+	SS/80	1024	832	20	224	204
Cor^{FO}	SS/80	2048	1696	22	116	114
Ours	SS/80	2048	1024	26	26	210 (130[c])

k security parameter (or security level)
[a] The time unit is defined as the cost of a general exponentiation (i.e., point multiplication) on a random MNT curve at the 80-bit security level
[b] Only KEM part is considered and the output size of hash function is calculated as 160 bits
[c] Obtained when a ratio of two pairings is calculated as 1.2 pairing

gular (SS) curves at the 80-bit security level. Table 4 shows the comparison result, which demonstrates that our system is comparable to the other IBE systems in terms of all efficiency respects even when not considering security losses caused by the security reductions. In particular, compared to AFG^+ and Cor^{FO} systems, ours has a remarkable advantage in terms of encryption, which is about 8.6 times faster than AFG^+ and about 4.5 times faster than Cor^{FO}.

For a fairer comparison, we try to compensate for such security losses by boosting the security parameter of the underlying assumptions. There is no general rule of such compensation, but we might be able to use conjectures that were made by Bellare and Rogaway [7] for advantage functions of various block ciphers. For instance, the advantage of DES with 128-bit keys was conjectured as (roughly speaking) $\mathbf{Adv}^{\mathrm{CPA}}_{\mathrm{AES}} \leq c/2^{128}$ for some constant c. A similar approach can be made for advantages of IBE systems, so that we want to make $\mathbf{Adv}^{\mathrm{CCA}}_{\mathcal{IBE},\mathcal{A}} \leq c/2^{80}$ at the 80-bit security level. In that case, Eq. 6 tells us that we have to make $\mathbf{Adv}^{\mathrm{Assumption}}_{\mathcal{B}} \leq c/2^{130}$ when considering that the security loss is bounded under 2^{50}. For simplicity, we assume that the reduction bounds in both BF and SK systems are 2^{50} (although they are much larger than it). Under these assumptions, the actual security parameter must be of 130 bits in size (approximately) to gain the real system security of an IBE system at the desired 80-bit security level. To accomplish this, we consider that the BF, BB_1, and SK systems are instantiated in MNT curves at the 128-bit security level, whereas AFG^+, Cor^{FO}, and ours are based on MNT curves at the 80-bit security level. Also, we assume that the output length of all hash functions (except the MapToPoint hash) is 160 bits and the MapToPoint hash function necessary for BF, AFG^+, Cor^{FO}, and ours maps an identity to a group element in $\mathbb{G}_2$. Table 5 shows the comparison result between those CCA-secure IBE systems. Compared to BF, SK, and BB_1 at the 128-bit security level, ours gives an advantage in terms of decryption, because decryption in ours is at least 6 times faster than those systems. Compared to AFG^+ and Cor^{FO} systems at the 80-bit security level, our IBE system provides at least about 1.6 times faster encryption than the others. However, AFG^+ system gives at least about 2.4 times shorter size of ciphertext (considering only KEM part) than Cor^{FO} and ours, and Cor^{FO} system provides at least about 1.8 times faster decryption than AFG^+ and ours (when a ratio of two pairings is considered as 2 pairings). It is therefore unclear which of the (asymmetric) DBDH-based IBE systems with tight security reduction must be chosen

Table 5 Estimated calculation times for CCA-secure IBE systems at corrected 80-bit security level

	Curves / k	Overheads (bits)		Relative timings[a]		
		PP	CT[b]	KGen	Enc	Dec
BF	MNT/128	1024	768	1000	2006	1503
BB_1	MNT/128	4608	1280	200	39	3033 (1833[c])
SK	MNT/128	4096	768	100	36	1506
AFG^+	MNT/80	342	491	80	280.4	200.4
Cor^{FO}	MNT/80	1368	1357	88	142.4	112.2
Ours	MNT/80	2223	1197	96.2	90.2	224.4 (144.4[c])

k security parameter (or security level)
[a] The time unit is defined as the cost of a general exponentiation (i.e., point multiplication) on a random MNT curve at the 80-bit security level
[b] Only KEM part is considered and the output size of hash function is calculated as 160 bits
[c] Obtained when a ratio of two pairings is calculated as 1.2 pairing

in general. It depends on the context of which efficiency factor out of encryption, decryption, and ciphertext size is taken into account significantly.

6 Discussion

6.1 On extension for hierarchical IBE system

In a hierarchical IBE (HIBE) system [31,35], a user's identity ID can be hierarchically scalable by delegating a private key $\mathsf{sk}_{\mathsf{ID}}$ to lower-level identities. For instance, a user with identity ID_1 can generate a private key $\mathsf{sk}_{\mathsf{ID}'}$ for a lower-level identity $\mathsf{ID}' = (\mathsf{ID}_1, \mathsf{ID}_2)$ using its own private key $\mathsf{sk}_{\mathsf{ID}_1}$. The reverse of key generation (i.e., from lower level to upper level) is not possible. This is called the 'delegation mechanism'. Using it, an HIBE system can be used for several applications including forward-secure encryption [18] and conversion for public key broadcast encryption [26]. In a security analysis for HIBE, an adversary is given the capability to request either private keys generated by a key generation center or ones delegated from upper-level identities of its choice.

One may wonder if our IBE system can be extended for supporting hierarchical identities. As far as we know, the answer is no. Since the private key structure of our system has similarity to that of Waters' tag-based dual system encryption [47], it may seem possible that a similar extension method can be applied to ours. However, the problem occurs because of the 'locked' tag values associated with upper-level identities. In the security analysis of the resulting HIBE system, an adversary requests a private key for an identity $\mathsf{ID} = (\mathsf{ID}_1, \ldots, \mathsf{ID}_\ell)$. In order to generate a private key $\mathsf{sk}_{\mathsf{ID}}$, we have to use one of the hidden values[8] that are embedded into $\{H_i(\mathsf{ID}_i)\}$ for $i = 1, \ldots, \ell$. Assume we use a hidden value in $H_k(\mathsf{ID}_k)$ for $k \leq \ell$. In that case, the tag values corresponding to j for $j < k$ are chosen at random and mapped to H_2-query outputs in an appropriate sense. However, those random tag values are locked and cannot be changed into other different values. The adversary can still query private keys for upper-level identities, e.g., $\mathsf{ID}' = (\mathsf{ID}_1, \ldots, \mathsf{ID}_{k-1})$, in which $\mathsf{sk}_{\mathsf{ID}'}$ should be generated using those locked tags. Unfortunately, such a private key cannot be generated. One solution

[8] Those are γ_i values that appear in the H_1^{list} in the proof of Theorems 1 and 2.

would be to use hidden values for each level of hierarchy, but instead it can reveal all hidden tag values that must be secretly reserved for challenge ciphertext. We leave it as an open problem to build a hierarchical version from our IBE system.

Acknowledgments The authors would like to thank the reviewers for their helpful comments and suggestions for this paper. Jong Hwan Park was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (NRF-2013R1A1A2009524). Dong Hoon Lee was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MEST) (No. 2010-0029121).

References

1. Abdalla M., Bellare M., Catalano D., Kiltz E., Kohno T., Lange T., Malone-Lee J., Neven G., Paillier P., Shi H.: Searchable encryption revisited: consistency properties, relation to anonymous IBE, and extensions. In: CRYPTO'05, vol. 3621, pp. 205–222 (2005)
2. Agrawal S., Boneh D., Boyen X.: Efficient lattice (H)IBE in the standard model. In: EUROCRYPT'10, vol. 6110, pp. 553–572 (2010)
3. Agrawal S., Boneh D., Boyen X.: Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE. In: CRYPTO'10, vol. 6223, pp. 98–115 (2010)
4. Agrawal S., Freeman D.F., Vaikuntanathan V.: Functional encryption for inner product predicates from learning with errors. In: ASIACRYPT'11, vol. 7073, pp. 21–40 (2011)
5. Attrapadung N., Furukawa J., Gomi T., Hanaoka G., Imai H., Zhang R.: Efficient identity-based encryption with tight security reduction. In: CANS'06, vol. 4301, pp. 19–36 (2006)
6. Bellare M., Rogaway P.: The exact security of digital signatures—how to sign with RSA and Rabin. In: EUROCRYPT'96, vol. 1070, pp. 399–416 (1996)
7. Bellare M., Rogaway P.: Introduction to Modern Cryptography. University of California at San Diego (2005)
8. Bentahar K., Farshim P., Malone-Lee J., Smart N.P.: Generic constructions of identity-based and certificateless KEMs. J. Cryptol. **21**(2), 178–199 (2008)
9. Bethencourt J., Sahai A., Waters B.: Ciphertext-policy attribute-based encryption. In: IEEE Symposium on Security and Privacy 2007, pp. 321–334 (2007)
10. Boneh D., Boyen X.: Efficient selective-id secure identity-based encryption without random oracles. In: EUROCRYPT'04, vol. 3027, pp. 223–238 (2004)
11. Boneh D., Boyen X., Goh E.J.: Hierarchical identity based encryption with constant size ciphertext. In: EUROCRYPT'05, vol. 3494, pp. 440–456 (2005)
12. Boneh D, Crescenzo G.D., Ostrovsky R., Persiano G.: Public key encryption with keyword search. In: EUROCRYPT'04, vol. 3027, pp. 506–522 (2004)
13. Boneh D., Franklin M.K.: Identity-based encryption from the weil pairing. In: CRYPTO'01, vol. 2139, pp. 213–229 (2001)
14. Boneh D., Gentry C., Hamburg M.: Space-efficient identity based encryption without pairings. In: FOCS'07, pp. 647–657 (2007)
15. Boneh D., Sahai A., Waters B.: Functional encryption: definitions and challenges. In: TCC'11, vol. 6597, pp. 253–273 (2011)
16. Boyen X.: A tapestry of identity-based encryption: practical frameworks compared. Int. J. Appl. Cryptogr. **1**(1), 3–21 (2008)
17. Boyen X., Waters B.: Anonymous hierarchical identity-based encryption (without random oracles). In: CRYPTO'06, vol. 4117, pp. 290–307 (2006)
18. Canetti R., Halevi S., Katz J.: A forward-secure public-key encryption scheme. In: EUROCRYPT03, vol. 2656, pp. 255–271 (2003)
19. Cash D, Hofheinz D, Kiltz E., Peikert C.: Bonsai trees, or how to delegate a lattice basis. In: EUROCRYPT10, vol. 6110, pp. 523–552 (2010)
20. Chen L., Cheng Z.: Security proof of Sakai-Kasahara's identity-based encryption scheme. In: IMA'05, vol. 3796, pp. 442–459 (2005)
21. Chen J., Wee H.: Fully, (almost) tightly secure ibe and dual system groups. In: CRYPTO'13, vol. 8043, pp. 435–460 (2013)
22. Chen L., Cheng Z., Malone-Lee J., Smart N.P.: An efficient ID-kem based on the Sakai-Kasahara key construction. IEE Proc. Inf. Secur. **153**(1), 19–26 (2006)

23. Cocks C.: An identity based encryption scheme based on quadratic residues. In: IMA'01, vol. 2260, pp. 360–363 (2001)
24. Coron J.S.: A variant of Boneh-Franklin IBE with a tight reduction in the random oracle model. Des. Codes Cryptogr. **50**(1), 115–133 (2009)
25. Cramer R., Shoup V.: Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. SIAM J. Comput. **33**(1), 167–226 (2004)
26. Dodis Y., Fazio N.: Public key broadcast encryption for stateless receivers. In: DRM Workshop 2002, vol. 2696, pp. 61–80 (2002)
27. Fujisaki E., Okamoto T.: Secure integration of asymmetric and symmetric encryption schemes. In: CRYPTO'99, vol. 1666, pp. 537–554 (1999)
28. Galindo D.: Boneh-Franklin identity based encryption revisited. In: ICALP'05, vol. 3580, pp. 791–802 (2005)
29. Gentry C.: Practical identity-based encryption without random oracles. In: EUROCRYPT06, vol. 4004, pp. 445–464 (2006)
30. Gentry C., Halevi S.: Hierarchical identity based encryption with polynomially many levels. In: TCC'09, vol. 5444, pp. 437–456 (2009)
31. Gentry C., Silverberg A.: Hierarchical id-based cryptography. In: ASIACRYPT'02, vol. 2501, pp. 548–566 (2002)
32. Gentry C., Peikert C., Vaikuntanathan V.: Trapdoors for hard lattices and new cryptographic constructions. In: STOC'08, pp. 197–206. ACM (2008)
33. Goh E.J., Jarecki S.: A signature scheme as secure as the diffie-hellman problem. In: EUROCRYPT'03, vol. 2656, pp. 401–415 (2003)
34. Goyal V., Pandey O., Sahai A., Waters B.: Attribute-based encryption for fine-grained access control of encrypted data. In: ACM-CCS'06, pp. 89–98. ACM (2006)
35. Horwitz J., Lynn B.: Toward hierarchical identity-based encryption. In: EUROCRYPT'02, vol. 2332, pp. 466–481 (2002)
36. Katz J., Wang N.: Efficiency improvements for signature schemes with tight security reductions. In: ACM-CCS'03, pp. 155–164. ACM (2003)
37. Kiltz E.: On the limitations of the spread of an IBE-to-PKE transformation. In: PKC'06, vol. 3958, pp. 274–289 (2006)
38. Kiltz E., Galindo D.: Direct chosen-ciphertext secure identity-based key encapsulation without random oracles. In: ACISP'06, vol. 4058, pp. 336–347 (2006)
39. Kiltz E., Vahlis Y.: CAA2 secure IBE: standard model efficiency through authenticated symmetric encryption. In: CT-RSA'08, vol. 4964, pp. 221–238 (2008)
40. Lewko A., Sahai A., Waters B.: Revocation systems with very small private keys. In: IEEE Symposium on Security and Privacy 2010, pp. 273–285 (2010)
41. Lewko A., Waters B.: New techniques for dual system encryption and fully secure HIBE with short ciphertexts. In: TCC'10, vol. 5978, pp. 455–579 (2010)
42. Sahai A., Waters B.: Fuzzy identity-based encryption. In: EUROCRYPT'05, vol. 3494, pp. 457–473 (2005)
43. Sakai R., Kasahara M.: Id based cryptosystems with pairing on elliptic curve. IACR Crypto. **54** (2003)
44. Seo J.H., Kobayashi T., Ohkubo M., Suzuki K.: Anonymous hierarchical identity-based encryption with constant size ciphertexts. In: PKC'09, vol. 5443, pp. 215–234 (2009)
45. Shamir A.: Identity-based cryptosystems and signature schemes. In: CRYPTO'84, vol. 196, pp. 47–53 (1984)
46. Waters B.: Efficient identity-based encryption without random oracles. In: EURO-CRYPT'05, vol. 3494, pp. 114–127 (2005)
47. Waters B.: Dual system encryption: realizing fully secure ibe and hibe under simpleassumptions. In: CRYPTO'09, vol. 5677, pp. 619–636 (2009)