Constant-Size Ciphertext HIBE From Asymmetric Pairings Using the Dual-System Technique

Somindu C. Ramanna and Palash Sarkar

Applied Statistics Unit
Indian Statistical Institute
203, B.T. Road, Kolkata
India 700108.
e-mail: {somindu_r,palash}@isical.ac.in

Abstract. We present a new hierarchical identity based encryption (HIBE) scheme with constant-size ciphertext that can be implemented using the most efficient bilinear pairings, namely, Type-3 pairings. Security argument is based on the dual-system technique of Waters. As a result, security can be proved under adaptive-identity attack and the security does not degrade with increase in the depth of the HIBE. The HIBE is obtained by extending an asymmetric pairing based IBE scheme due to Lewko and Waters. The extension follows an approach earlier used by Boneh, Boyen and Goh. The resulting HIBE is the only known scheme using Type-3 pairings achieving constant-size ciphertext and security against adaptive-identity attacks without random oracles.
Keywords: identity-based encryption(IBE), constant-size ciphertext hierarchical IBE, asymmetric pairings, dual-system encryption

1 Introduction

The notion of identity-based encryption (IBE) was introduced by Shamir [21] and the first IBE schemes appeared later [11, 3]. For an IBE scheme to function, a central authority called a private key generator (PKG) is required. The task of the PKG is to generate and securely distribute decryption keys corresponding to identities of different users. The notion of hierarchical IBE (HIBE) was proposed by [14, 15] to reduce the workload of the private key generator (PKG). In a HIBE scheme, the PKG is allowed to delegate the key generation ability to lower-level entities. As a result, an individual user can conveniently obtain a decryption key from a lower-level entity instead of obtaining it from the PKG.

Asymmetric Pairings: Practical (H)IBE schemes are built using bilinear pairings which maps $\mathbb{G}_1 \times \mathbb{G}_2$ to $\mathbb{G}_T$, where $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$ are groups of the same order. The only known examples of such maps arise by suitably choosing $\mathbb{G}_1$ and $\mathbb{G}_2$ to be groups of elliptic curve points and $\mathbb{G}_T$ to be a subgroup of the multiplicative group of a finite field. From an implementation point of view, it is most efficient to use bilinear maps where the (common) group order is prime and it is computationally difficult to find an isomorphism from $\mathbb{G}_1$ to $\mathbb{G}_2$ or vice versa. Such pairings are called Type-3 pairings [5, 23, 13]. Less efficient alternatives are when $\mathbb{G}_1$ and $\mathbb{G}_2$ are same (called Type-1 pairings) or when the common group order is a composite number (called composite-order pairings). IBE or HIBE schemes based on Type-3 pairings would have the fastest algorithms and the most compact representations of parameters.

Constant-Size Ciphertext HIBE: In a HIBE, identities consist of tuples of varying lengths. Encryption of a message is done for a specified identity tuple. In many HIBE schemes, as the

length of the identity tuple increases, so does the length of the resulting ciphertext. Consequently, the bandwidth requirement for communicating the ciphertext also increases. For an individual user, this may actually lead to a disincentive for obtaining decryption keys from lower-level entities. The net result may be in opposition to the basic motivation for a HIBE which is the ability to off-load work to lower-level entities.

The solution to this issue is to require that the ciphertext size is independent of the length of the identity tuple. Then, irrespective of the length of the identity tuple, the bandwidth required for the ciphertext would be the same. Such a scheme is called a constant-size ciphertext HIBE. The first such HIBE scheme was proposed by Boneh, Boyen and Goh [2]. While the scheme itself is quite elegant, its proof of security was in a very restricted attack model, the so-called selective-identity model. Lewko and Waters [16] provided the first constant-size ciphertext HIBE scheme which is secure against the usual adaptive-identity attacks. The drawback, however, was that the scheme in [16] used pairings on composite order groups and could not be instantiated with the more efficient Type-3 pairings.

Our Contributions: Our main motivation in this work is to obtain a constant-size ciphertext HIBE which can be implemented using Type-3 pairings. This allows the benefits of having constant-size ciphertexts to be combined with the efficiency benefits of using Type-3 pairings. These efficiency considerations are attained while retaining the usual provable guarantees, namely security against adaptive-identity attacks, use of static hardness assumptions, no degradation of security with increase in the depth of the HIBE and the avoidance of random oracles.

The nice provable properties can be achieved by using the extremely useful idea of dual-system encryption introduced by Waters [25]. This technique was used by Lewko and Waters [16] to construct an IBE and a HIBE scheme based on composite-order pairings. The authors in [16] went on to convert their composite-order pairing based IBE scheme to one which can be instantiated using Type-3 pairings. However, no such conversion was done for the HIBE scheme in [16] and the authors do not make any remark on whether this can be done or how difficult it would be to do so.

The starting point of our work is the Type-3 pairing based IBE scheme in [16] which we call LW-IBE. We extend LW-IBE to a constant-size ciphertext HIBE using Type-3 pairings. Unlike that of [16], we do not take the path of composite-order pairings. Instead we directly extend the LW-IBE scheme to a HIBE by suitably adapting an earlier identity hashing technique used by Boneh, Boyen and Goh [2]. One complication in doing so arises from combining the technique from [2] with that of dual-system technique [25]. In the dual-system technique, one has to appropriately define semi-functional components which have to be generated by the simulator while handling key extraction queries in the security reduction. Extending the decryption key of LW-IBE to the decryption key of a HIBE in a straightforward manner does not work. Our way of tackling this is to add additional components to the decryption key. On the face of it, this complicates the key generation and delegation mechanisms. However, somewhat counter-intuitively, adding this extra level of complication allows the security reductions to go through.

Table 1 compares our HIBE scheme with previously proposed constant-size ciphertext HIBE schemes. For reference, we also provide the details of the Waters 2009 HIBE scheme obtained using the dual-system technique. Even though this does not have constant-size ciphertexts, it is the first HIBE scheme obtained using the dual-system technique. In the table, n is the maximum depth of the HIBE and ℓ is the length of the identity vector. Columns two to five provide the number

of group elements respectively in the public parameters, the master secret key, ciphertexts and decryption keys. For the previous HIBE schemes shown, $\mathbb{G}_1 = \mathbb{G}_2$. In contrast, for the HIBE in this work (the last row), $\mathcal{PP}$ and ciphertexts consist of elements of $\mathbb{G}_1$ and $\mathcal{MSK}$ and decryption keys consists of elements of $\mathbb{G}_2$. In all cases, $\mathcal{PP}$ and the ciphertext have a single element of $\mathbb{G}_T$ which is not shown in the count. The count of group elements for $\mathcal{MSK}$ in the new HIBE is given as a pair. This is to denote that only one group element is required to be kept secret by the PKG; the other $(n+5)$ elements are required for key delegation and can be maintained publicly by the PKG. The quantity q is the number of key extraction queries made by an adversary.

From the table one can see that the Lewko-Waters HIBE (LW-HIBE) and the one given here are the only two HIBEs which have constant-size ciphertexts and achieve adaptive-id security (and also have security degradation of q). In absolute terms, the number of group elements required in the LW-HIBE is less than that required in the new HIBE scheme. However, only counting group elements is not a proper comparison. One has to consider the actual size for representing a single group element at a desired security level.

For concreteness, let us consider a security level of 128 bits. For Type-3 pairings, using Table-2 of [4], elements of $\mathbb{G}_1$ and $\mathbb{G}_2$ can be represented using 257 and 513 bits respectively. In contrast, the order of $\mathbb{G}_1 = \mathbb{G}_2$ in [16] is a product of three primes. The basic security requirement is that this group order should be hard to factor. To attain 128-bit security level, the length of the bit representation of the group order should be about 3000 bits (or more). So, for [16], the length of representations of elements of $\mathbb{G}_1$ (and $\mathbb{G}_2$) will be about 3000 bits. This is about 12 times (resp. 6 times) more than the length of bit representation of elements of $\mathbb{G}_1$ (resp. $\mathbb{G}_2$) using Type-3 pairings. The wide difference in the length of representations of group elements more than adequately compensates for the absolute number of group elements in LW-HIBE being lesser than that in the newly proposed HIBE scheme.

For example, ciphertexts in the new HIBE consist of 6 elements of $\mathbb{G}_1$ which is about 1500 bits whereas ciphertexts in the LW-HIBE will be about 9000 bits (3 elements each having length about 3000 bits). Similar considerations apply to public paramters ($\mathcal{PP}$), master secret key ($\mathcal{MSK}$) and decryption keys.

Scheme	$\#\mathcal{PP}$	$\#\mathcal{MSK}$	#cpr	#key	Security model	Pairing	Assumptions	Degradation
BBG-HIBE	$n+4$	1	2	$n-\ell+2$	selective-id	Type-1	D-wBDHI	1
Waters 2009	$2n+14$	2	$\ell+8$	$2\ell+7$	adaptive-id	Type-1	DLin,DBDH	q^2
Lewko-Waters	$n+3$	1	2	$n-\ell+2$	adaptive-id	Composite	Subgroup decision	q
This work	$3n+6$	$(1, n+5)$	6	$3n-3\ell+6$	adaptive-id	Type-3	LW1,LW2,DBDH	q

Table 1. Comparison of constant-size ciphertext HIBE schemes along with Waters [25] HIBE scheme.

Previous and Related Works: The work in [3] led to a great deal of research in constructions of IBE and HIBE schemes. We mention a few of these. For a more detailed description see [9].

Boneh and Boyen [1] presented a HIBE scheme using important algebraic ideas which were found very useful later. The HIBE itself was secure in a weaker model. Waters in 2005 [24] described an important IBE scheme which could be proved secure against adaptive-identity attacks without

using random oracles. This work also sketched a HIBE scheme. Variants of the IBE scheme were given in [6, 19] and a variant of the Waters-HIBE scheme was given in [7, 10].

The dual-system technique introduced by Waters [25] proved useful for constructing IBE, HIBE and related cryptographic primitives [16–18]. Waters description in [25] used Type-1 pairings. In a recent work [20], Waters IBE and related primitives have been simplified and it was shown how to instantiate them using Type-3 pairings.

As mentioned earlier, the first constant-size ciphertext HIBE was given by Boneh, Boyen and Goh [2]. This was secure against selective-identity attacks. Using the dual-system technique of Waters, a constant-size ciphertext HIBE was given in [16] using composite-order pairings.

2 Constant-Size Ciphertext HIBE

Section A provides a discussion on bilinear pairings and hardness assumptions in pairing groups, definitions and security models for IBE and HIBE followed by a desciption of Lewko-Waters IBE scheme.

In this section, we provide a HIBE scheme resulting from a BBG-type extension of the LW IBE scheme. A straightforward BBG-type extension would cause problems in simulating key extraction queries. We introduce some new elements to overcome this problem. The construction is based on a Type-3 prime-order pairing with group order p. Identities are variable length tuples of elements from ${\mathbb{Z}_p}^*$. Note that in the case of the IBE scheme, identities are elements of $\mathbb{Z}_p$. For moving to HIBE, it is necessary to remove 0 as a possible identity component. We discuss this in more details later.

2.1 Construction

Set-Up: Let n denote the maximum depth of the HIBE. Choose random generators $P_1 \in \mathbb{G}_1$ and $P_2 \in \mathbb{G}_2$; random elements $Q_{1,1}, \ldots, Q_{1,n}, U_1 \in \mathbb{G}_1$ and $Q_{2,1}, \ldots, Q_{2,n}, U_2 \in \mathbb{G}_2$ such that $Q_{2,j} \sim Q_{1,j}$ for all $1 \leq j \leq n$ and $U_2 \sim U_1$. Let $F_2 \in \mathbb{G}_2$ be chosen at random and v, v' be chosen randomly from $\mathbb{Z}_p$. Set $V_2 = vF_2$, $V_2' = v'F_2$. Pick α, a at random from $\mathbb{Z}_p$. Set $\tau = v + av'$ so that $\tau F_2 = V_2 + aV_2'$.

$\mathcal{PP}$: $(P_1, Q_{1,1}, \ldots, Q_{1,n}, U_1, aP_1, aQ_{1,1}, \ldots, aQ_{1,n}, aU_1, \tau P_1, \tau Q_{1,1}, \ldots, \tau Q_{1,n}, \tau U_1, e(P_1, P_2)^\alpha)$.
$\mathcal{MSK}$: $(\alpha P_2, P_2, V_2, V_2', F_2, Q_{2,1}, \ldots, Q_{2,n}, U_2)$.

The only quantity of the $\mathcal{MSK}$ which is required to be kept secret by the PKG is αP_2. The remaining components of the $\mathcal{MSK}$ are not required to be kept secret. These are required for key delegation and can be maintained by the PKG on its public database. So, in practice the PKG would choose to make both $\mathcal{PP}$ and the tuple $(P_2, V_2, V_2', F_2, Q_{2,1}, \ldots, Q_{2,n}, U_2)$ public. During encryption, only $\mathcal{PP}$ would be downloaded, whereas for key delegation the tuple $(P_2, V_2, V_2', F_2, Q_{2,1}, \ldots, Q_{2,n}, U_2)$ would be downloaded.

Encrypt$(M, \overrightarrow{\mathsf{id}} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathcal{PP})$: Choose random s from $\mathbb{Z}_p$. For an identity vector $\overrightarrow{\mathsf{id}} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$, let $H_i(\overrightarrow{\mathsf{id}}) = \mathsf{id}_1 Q_{i,1} + \cdots + \mathsf{id}_\ell Q_{i,\ell} + U_i$ for $i = 1, 2$. The ciphertext is given by $\mathcal{C} = (C_0, C_{1,1}, C_{1,2}, C_{1,3}, C_{2,1}, C_{2,2}, C_{2,3})$ where the elements are computed as follows.

$C_0 = M \times e(P_1, P_2)^{\alpha s}$,
$C_{1,1} = sH_1(\overrightarrow{\mathsf{id}})$, $C_{1,2} = asH_1(\overrightarrow{\mathsf{id}})$, $C_{1,3} = -\tau sH_1(\overrightarrow{\mathsf{id}})$
$C_{2,1} = sP_1$, $C_{2,2} = asP_1$, $C_{2,3} = -\tau sP_1$

KeyGen($\overrightarrow{\mathsf{id}} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathcal{MSK}, \mathcal{PP}$): Choose random r_1, r_2, t from $\mathbb{Z}_p$. The secret key for $\overrightarrow{\mathsf{id}}$, $\mathcal{SK}_{\overrightarrow{\mathsf{id}}}$ is given by $(K_{i,1}, K_{i,2}, K_{i,3}, D_{j,1}, D_{j,2}, D_{j,3})$ for $i = 1, 2$ and $j = \ell + 1, \ldots, n$. So, the key consists of $6 + 3(n - \ell)$ group elements. The elements are computed as follows.

$K_{1,1} = tP_2 + r_1V_2$, $K_{1,2} = r_1V_2'$, $K_{1,3} = r_1F_2$
$K_{2,1} = \alpha P_2 + tH_2(\overrightarrow{\mathsf{id}}) + r_2V_2$, $K_{2,2} = r_2V_2'$, $K_{2,3} = r_2F_2$
$D_{j,1} = tQ_{2,j} + r_jV_2$, $D_{j,2} = r_jV_2'$, $D_{j,3} = r_jF_2$ for $\ell + 1 \leq j \leq n$.

The portion of the key which consists of the K's is used for actual decryption. This part is the same as the key for the IBE scheme with the only change coming in the definition of the identity hash $H_2(\overrightarrow{\mathsf{id}})$. A straightforward extension of the IBE decryption key would have, for each j, only one D-component D_j to be equal to $tQ_{2,j}$. This would be required for creating the identity-hash for the next level delegation. However, such a direct extension does not work. In the security analysis, relating a decryption key to the problem instance in the second reduction does not work. The solution is to have a V_2 part in the D-component. But, then this requires separate randomisers $r_{\ell+1}, \ldots, r_n$ and also two extra components for cancellation.

Delegate($\overrightarrow{\mathsf{id}} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathcal{SK}_{\overrightarrow{\mathsf{id}}}, \mathsf{id}_{\ell+1}, \mathcal{PP}, (P_2, V_2, V_2', F_2, Q_{2,1}, \ldots, Q_{2,n}, U_2)$):

Note that the tuple $(P_2, V_2, V_2', F_2, Q_{2,1}, \ldots, Q_{2,n}, U_2)$ is required for key delegation and is made available publicly by the PKG. Define the notation $\overrightarrow{\mathsf{id}} : \mathsf{id}_{\ell+1} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell, \mathsf{id}_{\ell+1})$, i.e., $\overrightarrow{\mathsf{id}} : \mathsf{id}_{\ell+1}$ denotes the tuple obtained by concatenating $\mathsf{id}_{\ell+1}$ to the end of $\overrightarrow{\mathsf{id}}$.

Choose random $r_1', r_2', r_{\ell+2}', \ldots, r_n', t'$ from $\mathbb{Z}_p$. The components of the key for the identity $\overrightarrow{\mathsf{id}} : \mathsf{id}_{\ell+1}$ are computed as follows.

$K_{1,1} \leftarrow K_{1,1} + r_1'V_2 + t'P_2$, $K_{1,2} \leftarrow K_{1,2} + r_1'V_2'$, $K_{1,3} \leftarrow K_{1,3} + r_1'F_2$,
$K_{2,1} \leftarrow K_{2,1} + \mathsf{id}_{\ell+1}D_{\ell+1,1} + t'H_2(\overrightarrow{\mathsf{id}} : \mathsf{id}_{\ell+1}) + r_2'V_2$,
$K_{2,2} \leftarrow K_{2,2} + r_2'V_2' + \mathsf{id}_{\ell+1}D_{\ell+1,2}$,
$K_{2,3} \leftarrow K_{2,3} + r_2'F_2 + \mathsf{id}_{\ell+1}D_{\ell+1,3}$,
$D_{j,1} \leftarrow D_{j,1} + t'Q_{2,j} + r_j'V_2$, $D_{j,2} \leftarrow D_{j,2} + r_j'V_2'$, $D_{j,3} \leftarrow D_{j,3} + r_j'F_2$ for $j = \ell + 2, \ldots, n$.

The above procedure essentially re-randomises all components of the key. As a result the distribution of a key obtained using delegation is the same as the distribution of a key obtained using the key generation procedure. To note the re-randomisation consider the following change of scalars for the modified key.

$t \leftarrow t + t'$;
$r_1 \leftarrow r_1 + r_1'$;
$r_2 \leftarrow r_2 + \mathsf{id}_{\ell+1}r_{\ell+1} + r_2'$;
$r_j \leftarrow r_j + r_j'$ for $j = \ell + 2, \ldots, n$.

Since there is no difference in the distribution of delegated keys and keys directly generated by the PKG, in the security reduction the earlier Gentry-Silverberg [14] security model can be used rather than the more complete Shi-Waters [22] security model which can take into account the difference in distribution arising due to different delegation paths.

Decrypt $(\mathcal{C}, \overrightarrow{\mathsf{id}} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathcal{SK}_{\overrightarrow{\mathsf{id}}}, \mathcal{PP})$: Decryption is done as follows.

$$M = C_0 \times \frac{e(C_{1,1}, K_{1,1})e(C_{1,2}, K_{1,2})e(C_{1,3}, K_{1,3})}{e(C_{2,1}, K_{2,1})e(C_{2,2}, K_{2,2})e(C_{2,3}, K_{2,3})}$$

As is typical in the dual-system technique, we next describe the semi-functional ciphertexts and keys. These will not be required in the actual scheme and will be generated by the simulator in the security games.

Semi-functional ciphertext: Let $C'_0, C'_{1,1}, C'_{1,2}, C'_{1,3}, C'_{2,1}, C'_{2,2}, C'_{2,3}$ be ciphertext elements normally generated by the **Encrypt** algorithm for message M and identity $\overrightarrow{\mathsf{id}}$. Let V'_1, F_1 be an element of $\mathbb{G}_1$ such that $V'_1 \sim V'_2$ and $F_1 \sim F_2$. Choose $\mu, \pi_C \in \mathbb{Z}_p$ at random. The semi-functional ciphertext generation algorithm will modify the normal ciphertext as: $C_0 = C'_0$, $C_{1,1} = C'_{1,1}$, $C_{2,1} = C'_{2,1}$ and

$$C_{1,2} = C'_{1,2} + \mu\pi_C F_1,\ C_{1,3} = C'_{1,3} - \mu\pi_C V'_1, C_{2,2} = C'_{2,2} + \mu F_1,\ C_{2,3} = C'_{2,3} - \mu V'_1.$$

Semi-functional key: Let $(K'_{i,1}, K'_{i,2}, K'_{i,3}, D_{j,1}, D_{j,2}, D_{j,3})$ for $i = 1, 2$ and $j = \ell + 1, \ldots, n$ be secret key components generated by the **KeyGen** algorithm for identity $\overrightarrow{\mathsf{id}} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$. Let $\gamma, \pi_K, \gamma_{\ell+1}, \ldots, \gamma_n$ be random elements chosen from $\mathbb{Z}_p$. The semi-functional key generation algorithm will modify the normal key as:

$$K_{1,1} = K'_{1,1} - a\gamma F_2,\ K_{1,2} = K'_{1,2} + \gamma F_2,$$
$$K_{2,1} = K'_{2,1} - a\gamma\pi_K F_2,\ K_{2,2} = K'_{2,2} + \gamma\pi_K F_2,$$
$$D_{j,1} = D'_{j,1} - a\gamma_j F_2,\ D_{j,2} = D'_{j,2} + \gamma_j F_2 \ \text{ for } \ell + 1 \le j \le n.$$

The rest of the components, i.e., $K_{1,3}, K_{2,3}, D_{j,3}$ for $j = \ell + 1, \ldots, n$ remain unchanged.

The security of this HIBE scheme is based on the hardnesses of LW1, LW2 and the DBDH problems. Details are provided in Section 2.2.

Excluding zero as an identity component: As mentioned above, the identity components for the HIBE scheme above are from $\mathbb{Z}_p{}^*$ whereas for the IBE scheme in Section A.5 identities are elements of $\mathbb{Z}_p$. Allowing identity components to be zero in the HIBE scheme leads to a trivial attack as follows. Suppose that identities are from $\mathbb{Z}_p$ and the attacker $\mathcal{A}$ provides to the challenger, $\overrightarrow{\mathsf{id}}^*$ as the challenge identity. Then in the second key extraction phase, $\mathcal{A}$ could ask for a key for the identity vector $\overrightarrow{\mathsf{id}}^* : 0$ and hence essentially obtain a key for $\overrightarrow{\mathsf{id}}^*$ using which it can easily guess the message that was encrypted.

This is not an issue only with our HIBE scheme. The same is true of the BBG-HIBE scheme [2]. In fact, if one wishes to have zero as identity component in the BBG type scheme, then one has to pay the price of almost doubling the number of public parameters (see [8]). In the Lewko-Waters

paper [16], for the case of IBE it is mentioned that the identities are elements of $\mathbb{Z}_p$ (or $\mathbb{Z}_N$ for the composite order pairing based IBE). The description of the HIBE in [16] does not explicitly rule out the possibility of having zero as identity component. If identity components for the HIBE in [16] are allowed to be 0, then the above mentioned trivial attack applies. So, to get a secure scheme, the identity components of the HIBE in [16] should be elements of $\mathbb{Z}_N^*$.

2.2 Security Proof

We prove security of the HIBE scheme in the adaptive-identity security model proposed by Gentry-Silverberg [14]. The security argument for the scheme proceeds as before through $q+3$ games - Game_{real},$\text{Game}_0, \ldots, \text{Game}_q$ and Game_{final} where Game_{real} is the real HIBE security game. Let X_{real}, X_k for $k = 0, 1, \ldots, q$ and X_{final} be defined as earlier.

For the proof it will be convenient to use the following short-hand: Denote by $h(\overrightarrow{\mathsf{id}})$ the sum $\sum_{j=1}^{\ell} y_{q_j}\mathsf{id}_j + y_u$ and by $g(\overrightarrow{\mathsf{id}})$ the sum $\sum_{j=1}^{\ell} \lambda_j \mathsf{id}_j + \nu$, where $y_{q_1}, \ldots, y_{q_n}, y_u, \lambda_1, \ldots, \lambda_n, \nu$ are random elements of $\mathbb{Z}_p$ and will be defined in the proofs.

The simulation technique extends that of Lewko and Waters. Public parameters and decryption keys have extra components in comparison to LW-IBE. We show that these components can be constructed. For the components which are already present in LW-IBE, the simulation method remains more or less unchanged.

Lemma 1. *If there exists an adversary $\mathcal{A}$ such that $\mathsf{Adv}^{\mathcal{A}}_{\text{Game}_{real}} - \mathsf{Adv}^{\mathcal{A}}_{\text{Game}_0} = \varepsilon$, then we can build an algorithm $\mathcal{B}$ having advantage ε in solving the* LW1 *problem.*

Proof. The simulator $\mathcal{B}$ receives the following instance of LW1

$$(F_1, bsF_1, sF_1, aF_1, ab^2F_1, bF_1, b^2F_1, asF_1, b^2sF_1, b^3F_1, b^3sF_1, F_2, bF_2, Z_1).$$

It then simulates the security game as described below.

Set-Up: $\mathcal{B}$ chooses random elements $\alpha, y_g, y'_v, y_{q_1}, \ldots, y_{q_n}, y_u, \lambda_1, \ldots, \lambda_n, \nu$ from $\mathbb{Z}_p$ and sets the following elements.

$$P_1 = b^2F_1 + y_gF_1, Q_{1,j} = \lambda_j(b^2F_1) + y_{q_j}F_1 \text{ for } 1 \le j \le n, U_1 = \nu(b^2F_1) + y_uF_1$$

$$V_2 = bF_2, V'_2 = y'_vF_2.$$

This implicitly sets $\tau = b + ay'_v$. The element τP_1 can be computed as $b^3F_1 + y'_v(ab^2F_1) + y_g(bF_1) + y_gy'_v(aF_1)$. The parameter $e(P_1, P_2)^\alpha$ is computed as $(e(b^3F_1 + y_g(bF_1), bF_2)e(P_1, y_gF_2))^\alpha$. The simulator computes the remaining components of the public parameter using α and gives the following public parameters to $\mathcal{A}$.

$$\mathcal{PP} = (P_1, Q_1, U_1, aP_1, aQ_{1,1}, \ldots, aQ_{1,n}, aU_1, \tau P_1, \tau Q_{1,1}, \ldots, \tau Q_{1,n}, \tau U_1, e(P_1, P_2)^\alpha).$$

Phases 1 and 2: $\mathcal{A}$ makes a number of key extract queries. $\mathcal{B}$ does not know P_2 and hence does not have the master secret. The secret key for a query $\overrightarrow{\mathsf{id}}$ is constructed as follows. $\mathcal{B}$ chooses

$r'_1, r'_2, r'_{\ell+1}, \ldots, r'_n, t \in \mathbb{Z}_p$ at random and sets $r_1 = r'_1 - tb$, $r_2 = r'_2 - (tg(\overrightarrow{\mathsf{id}}) + \alpha)b$ and $r_j = r'_j - t\lambda_j b$ for $j = \ell + 1, \ldots, n$.

$$K_{1,1} = ty_g F_2 + r'_1(bF_2),\ K_{1,2} = y'_v r_1 F_2,\ K_{1,3} = r_1 F_2$$

$$K_{2,1} = \alpha y_g F_2 + r'_2(bF_2) + th(\overrightarrow{\mathsf{id}})F_2,\ K_{2,2} = y'_v r_2 F_2,\ K_{2,3} = r_2 F_2$$
$$D_{j,1} = ty_{q_j} F_2 + r'_j(bF_2),\ D_{j,2} = y'_v r_j F_2,\ D_{j,3} = r_j F_2 \text{ for } \ell + 1 \le j \le n$$

Challenge: $\mathcal{B}$ receives the target identity $\overrightarrow{\mathsf{id}}^* = (\mathsf{id}^*_1, \ldots, \mathsf{id}^*_{\ell^*})$ and two messages M_0 and M_1 from $\mathcal{A}$. It chooses $\beta \in \{0, 1\}$ at random. $\mathcal{B}$ computes the ciphertext for M_β as follows.

$$C_0 = M_\beta \cdot \left(e(b^3 sF_1 + y_g(bsF_1), bF_2)e(b^2 sF_1 + y_g(sF_1), y_g F_2)\right)^\alpha = M_\beta \cdot e(P_1, P_2)^{\alpha s}$$

$$C_{1,1} = g(\overrightarrow{\mathsf{id}}^*)(b^2 sF_1) + h(\overrightarrow{\mathsf{id}}^*)(sF_1),\ C_{1,2} = g(\overrightarrow{\mathsf{id}}^*)Z_1 + h(\overrightarrow{\mathsf{id}}^*)(asF_1)$$
$$C_{1,3} = -g(\overrightarrow{\mathsf{id}}^*)(b^3 sF_1) - h(\overrightarrow{\mathsf{id}}^*)(bsF_1) - y'_v g(\overrightarrow{\mathsf{id}}^*)Z_1 - y'_v h(\overrightarrow{\mathsf{id}}^*)(asF_1)$$
$$C_{2,1} = b^2 sF_1 + y_g(sF_1),\ C_{2,2} = Z_1 + y_g(asF_1)$$
$$C_{2,3} = -b^3 sF_1 - y_g(bsF_1) - y'_v Z_1 - y'_v(asF_1).$$

$\mathcal{B}$ returns $\mathcal{C}^* = (C_0, C_{1,1}, C_{1,2}, C_{1,3}, C_{2,1}, C_{2,2}, C_{2,3})$ to $\mathcal{A}$.

Now if $Z_1 = ab^2 sF_1$ then the challenge ciphertext is normal; otherwise if Z_1 is a random element of $\mathbb{G}_1$ i.e., $Z_1 = (ab^2 s + \mu')F_1$ then the ciphertext is semi-functional. Here μ is implicitly set to μ' and π_C is set to $g(\overrightarrow{\mathsf{id}}^*)$. It is straightforward to verify the well-formedness of $C_{1,2}$,$C_{1,3}$,$C_{2,2}$ and $C_{2,3}$. Note that, to check whether $\mathcal{C}^*$ is semi-functional or not, $\mathcal{B}$ itself could try to decrypt it with a semi-functional key for $\overrightarrow{\mathsf{id}}^*$. However since aP_2 is not known to $\mathcal{B}$, it cannot create such a key.

Guess: The adversary returns its guess β' to $\mathcal{B}$.

If $\mathcal{C}^*$ is normal then $\mathcal{B}$ simulates Game_{real} and if it is semi-functional $\mathcal{B}$ simulates Game_0. Therefore if $\mathcal{A}$ is able to distinguish between Game_{real} and Game_0 i.e., if $\beta = \beta'$, then the simulator can solve the LW1 problem with advantage

$$\mathsf{Adv}^{\mathcal{B}}_{\text{LW1}} = |\Pr[X_{real}] - \Pr[X_0]| = \mathsf{Adv}^{\mathcal{A}}_{\text{Game}_{real}} - \mathsf{Adv}^{\mathcal{A}}_{\text{Game}_0} = \varepsilon.$$

□

Lemma 2. *If there exists an adversary $\mathcal{A}$ such that $\mathsf{Adv}^{\mathcal{A}}_{\text{Game}_{k-1}} - \mathsf{Adv}^{\mathcal{A}}_{\text{Game}_k} = \varepsilon$, then we can build an algorithm $\mathcal{B}$ having advantage ε in breaking the assumption* LW2.

Proof. Let $(F_1, dF_1, d^2F_1, bxF_1, dbxF_1, d^2xF_1, F_2, dF_2, bF_2, cF_2, Z_2)$ be the instance of LW2 that $\mathcal{B}$ receives.

Set-Up: $\mathcal{B}$ chooses random elements $\alpha, a, y_v, y_{q_1}, \ldots, y_{q_n}, y_u, \lambda_1, \ldots, \lambda_n, \nu \in_R \mathbb{Z}_p$ and sets the following elements. $P_1 = dF_1, P_2 = dF_2$, $Q_{1,j} = \lambda_j(dF_1) + y_{q_j}F_1$ for $1 \le j \le n$, $U_1 = \nu(dF_1) + y_u F_1$, $V_2 = -a(bF_2) + dF_2 + y_v F_2$ and $V'_2 = bF_2$ setting $v = -ab + d + y_v$, $v' = b$ and $\tau = d + y_v$. The element τP_1 can be computed as $\tau P_1 = d^2F_1 + y_v(dF_1)$. The parameters $\tau Q_{1,j}$ for $1 \le j \le n$ and τU_1 can be computed as $\tau Q_{1,j} = \lambda_j(d^2F_1) + y_{q_j}(dF_1) + y_v\lambda_j(dF_1) + y_v y_{q_j}F_1$ and $\tau U_1 =$

$\nu(d^2F_1) + y_u(dF_1) + y_v\nu(dF_1) + y_vy_uF_1$. The remaining parameters required to provide $\mathcal{PP}$ to $\mathcal{A}$ are computed using a, α and other elements of the problem instance.

Phases 1 and 2: The key extraction queries for identities $\overrightarrow{\mathsf{id}}_1, \ldots, \overrightarrow{\mathsf{id}}_q$ are answered in the following way. For $i < k$, a semi-functional key is returned and for $i > k$ a normal key is returned. This can be done since $\mathcal{B}$ has the master secret and also knows a.

For $i = k$, $\mathcal{B}$ computes the components of the key using the problem instance in the following manner. Let $\overrightarrow{\mathsf{id}}_k = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$. $\mathcal{B}$ chooses $t', r'_2, r'_{\ell+1}, \ldots, r'_n$ randomly from $\mathbb{Z}_p$.

$$K_{1,1} = t'P_2 - aZ_2 + y_v(cF_2),\ K_{1,2} = Z_2,\ K_{1,3} = cF_2$$

$$K_{2,1} = \alpha P_2 + t'(g(\overrightarrow{\mathsf{id}}_k)(dF_2) + h(\overrightarrow{\mathsf{id}}_k)F_2) + r'_2V_2 - ag(\overrightarrow{\mathsf{id}}_k)Z_2 + y_vg(\overrightarrow{\mathsf{id}}_k)(cF_2) - h(\overrightarrow{\mathsf{id}}_k)cF_2$$

$$K_{2,2} = r'_2V'_2 + g(\overrightarrow{\mathsf{id}}_k)Z_2,\ K_{2,3} = r'_2F_2 + g(\overrightarrow{\mathsf{id}}_k)(cF_2)$$

and for $j = \ell + 1, \ldots, n$, set

$$D_{j,1} = t'Q_{2,j} + r'_jV_2 - y_{q_j}(cF_2) - a\lambda_jZ_2 + y_v\lambda_j(cF_2)$$

$$D_{j,2} = r'_jV'_2 + \lambda_jZ_2,\ D_{j,3} = r'_jF_2 + \lambda_j(cF_2)$$

thus implicitly setting $r_1 = c$, $r_2 = r'_2 + g(\overrightarrow{\mathsf{id}}_k)c$, $t = t' - c$, $\pi_K = g(\overrightarrow{\mathsf{id}}_k)$ and $r_j = r'_j + \lambda_jc$ for $\ell + 1 \leq j \leq n$.

$\mathcal{B}$ returns $\mathcal{SK}_{\overrightarrow{\mathsf{id}}_k} = (K_{i,1}, K_{i,2}, K_{i,3}, D_{j,1}, D_{j,2}, D_{j,3})$, $i = 1, 2$ and $j = 1, 2, \ell + 1, \ldots, n$ as the key for $\overrightarrow{\mathsf{id}}_k$. If $Z_2 = bcF_2$ then the key for $\overrightarrow{\mathsf{id}}_k$ will be normal and if $Z_2 = (bc + \gamma')F_2$ the key will be semi-functional with $\gamma = \gamma'$, $\gamma_j = \lambda_j\gamma'$ for $\ell + 1 \leq j \leq n$. $\mathcal{B}$ could also check whether $\mathcal{SK}_{\overrightarrow{\mathsf{id}}_k}$ is normal or semi-functional by creating a semi-functional ciphertext for id_k. Since $V'_1 = bF_1$ is not available to $\mathcal{B}$, the only way of doing this will lead to π_C being the same as π_K (such a ciphertext can be constructed in a manner similar to creating the challenge ciphertext). The ciphertext-key pair will be nominally semi-functional and thus provides no information to $\mathcal{B}$.

Challenge: $\mathcal{B}$ receives two messages M_0, M_1 and a challenge identity $\overrightarrow{\mathsf{id}}^* = (\mathsf{id}^*_1, \ldots, \mathsf{id}^*_{\ell^*})$ during the challenge phase. It chooses $\beta \in_R \{0, 1\}$, generates the challenge ciphertext as shown below.

$$C_0 = M_\beta \cdot e(dbxF_1, dF_2)^\alpha$$

$$C_{1,1} = g(\overrightarrow{\mathsf{id}}^*)(dbxF_1) + h(\overrightarrow{\mathsf{id}}^*)(bxF_1)$$

$$C_{1,2} = ag(\overrightarrow{\mathsf{id}}^*)(dbxF_1) + ah(\overrightarrow{\mathsf{id}}^*)(bxF_1) - g(\overrightarrow{\mathsf{id}}^*)(d^2xF_1)$$

$$C_{1,3} = -y_vg(\overrightarrow{\mathsf{id}}^*)(dbxF_1) - h(\overrightarrow{\mathsf{id}}^*)(dbxF_1) - y_vh(\overrightarrow{\mathsf{id}}^*)(bxF_1)$$

$$C_{2,1} = dbxF_1,\ C_{2,2} = a(dbxF_1) - d^2xF_1,\ C_{2,3} = -y_v(dbxF_1).$$

This sets $s = bx$, $\mu = -d^2x$ and $\pi_C = g(\overrightarrow{\mathsf{id}}^*)$.

Since $\lambda_1, \ldots, \lambda_n$ and ν are chosen uniformly at random from $\mathbb{Z}_p$, $\pi_K = \lambda_1\mathsf{id}_1 + \cdots + \lambda_\ell\mathsf{id}_\ell + \nu$ and $\pi_C = \lambda_1\mathsf{id}^*_1 + \cdots + \lambda_{\ell^*}\mathsf{id}^*_{\ell^*} + \nu$ are independent and uniformly distributed. Now $\mathcal{A}$ will be

able to distinguish between Game_{k-1} and Game_k if it can decide whether $\mathcal{SK}_{\overrightarrow{\mathsf{id}}_k}$ is normal or semi-functional. In this case, $\mathcal{B}$ can break the assumption LW2 with advantage

$$\mathsf{Adv}^{\mathcal{B}}_{\text{LW2}} = |\Pr[X_{k-1}] - \Pr[X_k]| = \mathsf{Adv}^{\mathcal{A}}_{\text{Game}_{k-1}} - \mathsf{Adv}^{\mathcal{A}}_{\text{Game}_k} = \varepsilon.$$

□

Lemma 3. *If there exists an adversary $\mathcal{A}$ such that $\mathsf{Adv}^{\mathcal{A}}_{\text{Game}_q} - \mathsf{Adv}^{\mathcal{A}}_{final} = \varepsilon$, then we can build an algorithm $\mathcal{B}$ having advantage ε in breaking the* DBDH *assumption.*

Proof. $\mathcal{B}$ receives $(F_1, aF_1, bF_1, sF_1, F_2, aF_2, bF_2, sF_2, Z_T)$ as an instance of the DBDH problem where $Z_T = e(F_1, F_2)^{abs}$ or $Z_T \in_R \mathbb{G}_T$.

Set-Up: With $y_g, y_v, y'_v, y_{q_1}, \ldots, y_{q_n}, y_u$ chosen at random from $\mathbb{Z}_p$, $\mathcal{B}$ sets the parameters as

$$P_1 = y_g F_1, P_2 = y_g F_2, aP_1 = y_g(aF_1), V_2 = y_v F_2, V'_2 = y'_v F_2,\ \tau P_1 = y_g y_v F_1 + y_g y'_v (aF_1)$$

$$Q_{1,j} = y_{q_j} F_1 \text{ for } 1 \le j \le n, U_1 = y_u F_1, e(P_1, P_2)^{\alpha} = e(aF_1, bF_2)^{{y_g}^2}$$

implicitly setting $\alpha = ab$ and $\tau = y_v + ay'_v$. The remaining parameters can be computed easily. $\mathcal{B}$ returns $\mathcal{PP}$ to $\mathcal{A}$.

Phases 1 and 2: When $\mathcal{A}$ asks for the secret key for the i'th identity $\overrightarrow{\mathsf{id}}_i = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$, $\mathcal{B}$ chooses at random $t, r_1, r_2, r_{\ell+1}, \ldots r_n \in \mathbb{Z}_p$ and $\gamma', \gamma, \gamma_{\ell+1}, \ldots, \gamma_n \in \mathbb{Z}_p$ and computes a semi-functional key for $\overrightarrow{\mathsf{id}}_i$ as follows.

$$K_{1,1} = tP_2 + r_1 V_2 - \gamma(aF_2),\ K_{1,2} = r_1 V'_2 + \gamma F_2,\ K_{1,3} = r_1 F_2$$

$$K_{2,1} = \gamma'(aF_2) + th(\overrightarrow{\mathsf{id}}_i)F_2 + r_2 V_2,\ K_{2,2} = r_2 V'_2 + y_g(bF_2) - \gamma' F_2,\ K_{2,3} = r_2 F_2,$$

$$D_{j,1} = tQ_{2,j} + r_j V_2 - \gamma_j(aF_2),\ D_{j,2} = r_j V'_2 + \gamma_j F_2,\ D_{j,3} = r_j F_2 \text{ for } \ell + 1 \le j \le n.$$

Here $\gamma' = by_g - \gamma\pi_K$ and is implicit in the construction. Observe that $\mathcal{B}$ does not know α and hence cannot create a normal key.

Challenge: $\mathcal{B}$ receives the challenge identity $\overrightarrow{\mathsf{id}}^* = (\mathsf{id}^*_1, \ldots, \mathsf{id}^*_{\ell^*})$ and two messages M_0 and M_1 from $\mathcal{A}$. It chooses $\beta \in \{0, 1\}$ and $\mu_1, \mu_2 \in \mathbb{Z}_p$ at random and generates a semi-functional challenge ciphertext as follows.

$$C_0 = M_\beta \times Z_T$$

$$C_{1,1} = h(\overrightarrow{\mathsf{id}}^*)sF_1,\ C_{1,2} = h(\overrightarrow{\mathsf{id}}^*)\mu_1 F_1,\ C_{1,3} = -y_v h(\overrightarrow{\mathsf{id}}^*)(sF_1) - y'_v h(\overrightarrow{\mathsf{id}}^*)\mu_1 F_1$$

$$C_{2,1} = y_g(sF_1),\ C_{2,2} = \mu_2 F_1,\ C_{2,3} = -y_g y_v(sF_1) - y'_v \mu_2 F_1$$

with $\mu_1 = as + \mu$, $\mu_2 = asy_g + \mu$ and $\pi_C = h(\overrightarrow{\mathsf{id}}^*)$.

The challenge ciphertext $\mathcal{C}^* = (C_0, C_{1,1}, C_{1,2}, C_{1,3}, C_{2,1}, C_{2,2}, C_{2,3})$ is returned to $\mathcal{A}$. If $Z_T = e(P_1, P_2)^{abs}$ then $\mathcal{C}^*$ will be a semi-functional encryption of M_β; if Z_T is a random element of $\mathbb{G}_T$ then $\mathcal{C}^*$ will be a semi-functional encryption of a random message. If $\mathcal{A}$ can identify whether the game simulated was Game_q or Game_{final}, then $\mathcal{B}$ will be able to decide whether $Z_T = e(P_1, P_2)^{abs}$ or not and hence break the DBDH assumption with advantage

$$\mathsf{Adv}^{\mathcal{B}}_{\text{DBDH}} = |\Pr[X_q] - \Pr[X_{final}]| = \mathsf{Adv}^{\mathcal{A}}_{\text{Game}_q} - \mathsf{Adv}^{\mathcal{A}}_{\text{Game}_{final}} = \varepsilon.$$

□

Theorem 1. *If the* LW1*,* LW2 *and* DBDH *assumptions hold, then no polynomial time adversary $\mathcal{A}$ making at most q key extraction queries can break the security of the HIBE in Section 2.1.*

Proof. Using lemmas 1, 2 and 3, we have for any polynomial time attacker $\mathcal{A}$,

$$\begin{aligned}\mathsf{Adv}^{\mathcal{A}}_{\mathrm{HIBE}} &\leq |\Pr[X_{real}] - \Pr[X_0]| + \sum_{k=1}^{q} (|\Pr[X_{k-1}] - \Pr[X_k]|) + |\Pr[X_q] - \Pr[X_{final}]| \\ &= \varepsilon_{\mathrm{LW1}} + q\varepsilon_{\mathrm{LW2}} + \varepsilon_{\mathrm{DBDH}}\end{aligned}$$

which is negligible in the security parameter κ. □

3 Conclusion

We have extended the Lewko-Waters IBE scheme using asymmetric pairings to a constant-size ciphertext HIBE. Security of the HIBE is based on the LW1, LW2 and DBDH assumptions. This HIBE is the first example of an adaptive-id secure, constant-size ciphertext HIBE which can be instantiated using Type-3 pairings. This improves upon the previous work by Lewko and Waters which used composite-order pairings to obtain adaptive-id secure constant-size ciphertext HIBE.

Acknowledgement

We thank the reviewers of Pairing 2012 for pointing out errors in the security proofs presented in an earlier version of this paper. We would also like to thank the program chairs for providing us an opportunity to correct the errors.

References

1. Dan Boneh and Xavier Boyen. Efficient selective-ID secure identity-based encryption without random oracles. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT*, volume 3027 of *Lecture Notes in Computer Science*, pages 223–238. Springer, 2004.
2. Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity-based encryption with constant size ciphertext. In Cramer [12], pages 440–456. Full version available at Cryptology ePrint Archive; Report 2005/015.
3. Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. *SIAM J. Comput.*, 32(3):586–615, 2003. Earlier version appeared in the proceedings of CRYPTO 2001.
4. Sanjit Chatterjee, Darrel Hankerson, Edward Knapp, and Alfred Menezes. Comparing two pairing-based aggregate signature schemes. *Des. Codes Cryptography*, 55(2-3):141–167, 2010.
5. Sanjit Chatterjee and Alfred Menezes. On cryptographic protocols employing asymmetric pairings – the role of ψ revisited. *Discrete Applied Mathematics*, 159(13):1311–1322, 2011.
6. Sanjit Chatterjee and Palash Sarkar. Trading time for space: Towards an efficient IBE scheme with short(er) public parameters in the standard model. In Dong Ho Won and Seungjoo Kim, editors, *ICISC*, volume 3935 of *Lecture Notes in Computer Science*, pages 424–440. Springer, 2005.
7. Sanjit Chatterjee and Palash Sarkar. Hibe with short public parameters without random oracle. In X. Lai and K. Chen, editors, *ASIACRYPT*, volume 4284 of *Lecture Notes in Computer Science*, pages 145–160. Springer, 2006. see also Cryptology ePrint Archive, Report 2006/279, http://eprint.iacr.org/.
8. Sanjit Chatterjee and Palash Sarkar. Constant size ciphertext hibe in the augmented selective-id model and its extensions. *J. UCS*, 13(10):1367–1395, 2007.

9. Sanjit Chatterjee and Palash Sarkar. *Identity-Based Encryption*. Springer, 2010.
10. Sanjit Chatterjee and Palash Sarkar. Practical hybrid (hierarchical) identity-based encryption schemes based on the decisional bilinear diffie-hellman assumption. CACR technical report, Report 2010/020, 2010. `http://www.cacr.math.uwaterloo.ca/tech_reports.html`.
11. Clifford Cocks. An identity-based encryption scheme based on quadratic residues. In Bahram Honary, editor, *IMA Int. Conf.*, volume 2260 of *Lecture Notes in Computer Science*, pages 360–363. Springer, 2001.
12. Ronald Cramer, editor. *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, volume 3494 of *Lecture Notes in Computer Science*. Springer, 2005.
13. Steven D. Galbraith, Kenneth G. Paterson, and Nigel P. Smart. Pairings for cryptographers. *Discrete Applied Mathematics*, 156(16):3113–3121, 2008.
14. Craig Gentry and Alice Silverberg. Hierarchical ID-based cryptography. In Yuliang Zheng, editor, *ASIACRYPT*, volume 2501 of *Lecture Notes in Computer Science*, pages 548–566. Springer, 2002.
15. Jeremy Horwitz and Ben Lynn. Toward hierarchical identity-based encryption. In Lars R. Knudsen, editor, *EUROCRYPT*, volume 2332 of *Lecture Notes in Computer Science*, pages 466–481. Springer, 2002.
16. Allison B. Lewko and Brent Waters. New techniques for dual system encryption and fully secure HIBE with short ciphertexts. In Daniele Micciancio, editor, *TCC*, volume 5978 of *Lecture Notes in Computer Science*, pages 455–479. Springer, 2010.
17. Allison B. Lewko and Brent Waters. Unbounded hibe and attribute-based encryption. In Kenneth G. Paterson, editor, *EUROCRYPT*, volume 6632 of *Lecture Notes in Computer Science*, pages 547–567. Springer, 2011.
18. Benoît Libert and Moti Yung. Adaptively secure non-interactive threshold cryptosystems. In Luca Aceto, Monika Henzinger, and Jiri Sgall, editors, *ICALP (2)*, volume 6756 of *Lecture Notes in Computer Science*, pages 588–600. Springer, 2011.
19. David Naccache. Secure and practical identity-based encryption. *IET Information Security*, 1(2):59–64, 2007.
20. Somindu C. Ramanna, Sanjit Chatterjee, and Palash Sarkar. Variants of Waters' dual system primitives using asymmetric pairings. Cryptology ePrint Archive, Report 2012/024, 2012. `http://eprint.iacr.org/`.
21. Adi Shamir. Identity-based cryptosystems and signature schemes. In G. R. Blakley and David Chaum, editors, *CRYPTO*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53. Springer, 1984.
22. Elaine Shi and Brent Waters. Delegating capabilities in predicate encryption systems. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz, editors, *ICALP (2)*, volume 5126 of *Lecture Notes in Computer Science*, pages 560–578. Springer, 2008.
23. Nigel P. Smart and Frederik Vercauteren. On computable isomorphisms in efficient asymmetric pairing-based systems. *Discrete Applied Mathematics*, 155(4):538–547, 2007.
24. Brent Waters. Efficient identity-based encryption without random oracles. In Cramer [12], pages 114–127.
25. Brent Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In Shai Halevi, editor, *CRYPTO*, volume 5677 of *Lecture Notes in Computer Science*, pages 619–636. Springer, 2009.

A Preliminaries

A.1 Bilinear pairings

A bilinear pairing is a map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ where $\mathbb{G}_1$,$\mathbb{G}_2$ are additive groups and $\mathbb{G}_T$ is a multiplicative group, all having the same group order, with the following properties.

1. *Bilinear:* For $P_1, Q_1 \in \mathbb{G}_1$ and $P_2, Q_2 \in \mathbb{G}_2$, the following holds:
   $e(P_1, P_2 + Q_2) = e(P_1, P_2)e(P_1, Q_2)$ and $e(P_1 + Q_1, P_2) = e(P_1, P_2)e(Q_1, P_2)$.
2. *Non-degenerate:* If $e(P_1, P_2) = 1_T$, the identity element of $\mathbb{G}_T$, then either P_1 is the identity of $\mathbb{G}_1$ or P_2 is the identity of $\mathbb{G}_2$.
3. *Efficiently computable:* The function e should be efficiently computable.

Three main types of pairings have been identified in the literature [23, 13].

Type-1 In this type, the groups $\mathbb{G}_1$ and $\mathbb{G}_2$ are the same.
Type-2 $\mathbb{G}_1 \neq \mathbb{G}_2$ and an efficiently computable isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$ is known.
Type-3 Here, $\mathbb{G}_1 \neq \mathbb{G}_2$ and no efficiently computable isomorphisms between $\mathbb{G}_1$ and $\mathbb{G}_2$ are known.

It has been reported [5, 23, 13] that from an implementation point of view, Type-3 pairings are the fastest to compute and further provide the most compact description of group elements. So, from a practical point of view it is of interest to have functionalities which can be instantiated using Type-3 pairings. This work is based on Type-3 pairings with all the groups having the same prime order.

A.2 Complexity Assumptions

We state here some hardness assumptions in the Type-3 setting that will be used in the security proofs.

Assumption LW1. Let $F_1 \in \mathbb{G}_1$ and $F_2 \in \mathbb{G}_2$ be random generators and let a, b, s be chosen randomly from $\mathbb{Z}_p$. Let $Y_1 \in_R \mathbb{G}_1$. For a probabilistic polynomial time (PPT) algorithm $\mathcal{A}$ that outputs 0 or 1, its advantage in solving the LW1 problem is given by

$$\begin{aligned}\mathsf{Adv}^{\mathcal{A}}_{\mathrm{LW1}} =& |\Pr[\mathcal{A}(F_1, bsF_1, sF_1, aF_1, ab^2F_1, bF_1, b^2F_1, asF_1, b^2sF_1, b^3F_1, b^3sF_1, F_2, bF_2, ab^2sF_1) = 1] \\ &- \Pr[\mathcal{A}(F_1, bsF_1, sF_1, aF_1, ab^2F_1, bF_1, b^2F_1, asF_1, b^2sF_1, b^3F_1, b^3sF_1, F_2, bF_2, Y_1) = 1]|\end{aligned}$$

The LW1 assumption is that for every PPT algorithm $\mathcal{A}$, $\mathsf{Adv}^{\mathcal{A}}_{\mathrm{LW1}}$ is negligible.

Assumption LW2. Let $F_1 \in \mathbb{G}_1$ and $F_2 \in \mathbb{G}_2$ be random generators. Let d, x_1, x_2 denote random elements of $\mathbb{Z}_p$. The LW2 problem is to decide, given

$$(F_1, dF_1, d^2F_1, bxF_1, dbxF_1, d^2xF_1, F_2, dF_2, bF_2, cF_2, Z_2),$$

whether $Z_2 = bcF_2$ or $Z_2 \in_R \mathbb{G}_2$. The LW2 assumption is that any PPT algorithm $\mathcal{A}$ has negligible advantage in solving the LW2 problem.

Decisional Bilinear Diffie-Hellman (DBDH) assumption. Let P_1, P_2 be random generators of $\mathbb{G}_1$,$\mathbb{G}_2$ respectively and let x_1, x_2, x_3 be random elements of $\mathbb{Z}_p$. Let Y_T be a random element of $\mathbb{G}_T$. For a PPT algorithm $\mathcal{A}$ that outputs either 0 or 1, define its advantage in solving the DBDH problem as follows.

$$\begin{aligned}\mathsf{Adv}^{\mathcal{A}}_{\mathrm{DBDH}} =& |\Pr[\mathcal{A}(P_1, x_1P_1, x_2P_1, x_3P_1, P_2, x_1P_2, x_2P_2, x_3P_2, e(P_1, P_2)^{x_1x_2x_3}) = 1] \\ &- \Pr[\mathcal{A}(P_1, x_1P_1, x_2P_1, x_3P_1, P_2, x_1P_2, x_2P_2, x_3P_2, Y_T) = 1]|\end{aligned}$$

The DBDH assumption is that for every PPT algorithm $\mathcal{A}$, $\mathsf{Adv}^{\mathcal{A}}_{\mathrm{DBDH}}$ is negligible.

A.3 Definitions of IBE and HIBE

An IBE scheme is defined by four probabilistic algorithms namely **Set-Up**, **KeyGen**, **Encrypt** and **Decrypt** . The algorithms have running time polynomial in a security parameter κ. The first two algorithms are run by the private key generator (PKG) and the other two by users.

- **Set-Up** generates the public parameters $\mathcal{PP}$ and the master secret key $\mathcal{MSK}$. $\mathcal{PP}$ is made public while $\mathcal{MSK}$ remains secret.
- **KeyGen** inputs an identity id and the master secret $\mathcal{MSK}$; outputs a secret key $\mathcal{SK}_{\mathsf{id}}$ for id.
- **Encrypt** takes as input an identity id and a message M and returns a ciphertext $\mathcal{C}$.
- **Decrypt** inputs a ciphertext $\mathcal{C}$ for an identity id and secret key $\mathcal{SK}_{\mathsf{id}}$; returns the encrypted message M or $\perp$ indicating failure.

A HIBE scheme consists of five probabilistic polynomial time algorithms - **Set-Up**, **Encrypt**, **KeyGen**, **Delegate** and **Decrypt** .

- **Set-Up** outputs the public parameters $\mathcal{PP}$ and the master secret $\mathcal{MSK}$.
- **KeyGen** inputs an identity vector $\overrightarrow{\mathsf{id}}$ and master secret $\mathcal{MSK}$ and outputs the secret key $\mathcal{SK}_{\overrightarrow{\mathsf{id}}}$ corresponding to $\overrightarrow{\mathsf{id}}$.
- **Encrypt** inputs an identity $\overrightarrow{\mathsf{id}}$, a message M and returns a ciphertext $\mathcal{C}$.
- **Delegate** takes as input a depth ℓ identity vector $\overrightarrow{\mathsf{id}} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$, a secret key $\mathcal{SK}_{\overrightarrow{\mathsf{id}}}$ and an identity $\mathsf{id}_{\ell+1}$; returns a secret key for the identity vector $(\mathsf{id}_1, \ldots, \mathsf{id}_{\ell+1})$.
- **Decrypt** inputs a ciphertext $\mathcal{C}$, an identity vector $\overrightarrow{\mathsf{id}}$, secret key $\mathcal{SK}_{\overrightarrow{\mathsf{id}}}$ and returns either the corresponding message M or $\perp$ indicating failure.

A.4 Security models

Here we provide the security models for IBE and HIBE schemes in the adaptive identity setting.

IBE: Security of an IBE scheme is modeled using a game between an adversary $\mathcal{A}$ and a challenger. Mentioned below are the different phases of the game.

Setup: The challenger inputs a security parameter κ and runs the **Set-Up** algorithm of the IBE and gives the public parameters to $\mathcal{A}$.

Phase 1: $\mathcal{A}$ makes a number of queries for secret keys of identities adaptively. The challenger receives an id from $\mathcal{A}$, computes the secret key corresponding to id and then returns a key for id to $\mathcal{A}$.

Challenge: $\mathcal{A}$ provides a challenge identity id^* and two equal length messages M_0 and M_1 to the challenger with the restriction that id^* should not have been queried in **Phase 1**. The challenger then chooses a bit β at random from $\{0, 1\}$ and returns an encryption of M_β to $\mathcal{A}$.

Phase 2: $\mathcal{A}$ issues more key extraction queries like in **Phase 1** but it is not allowed to request a key for id^*.

Guess: $\mathcal{A}$ outputs a bit β' as its guess for β.

$\mathcal{A}$ wins the above game if $\beta = \beta'$. The advantage of $\mathcal{A}$ in breaking the security of the IBE scheme is defined in terms of the probability of the event that $\beta = \beta'$ in the above game as shown below.

$$\mathsf{Adv}^{\mathcal{A}}_{\mathrm{IBE}} = \left| \Pr[\beta = \beta'] - \frac{1}{2} \right|$$

The IBE scheme is said to be secure if an probabilistic polynomial time (in security parameter κ) adversary has advantage (of winning the above game) negligible in κ.

HIBE: The security for a HIBE scheme is also modeled as a game between an adversary $\mathcal{A}$ and a simulator. The following are the different phases of the security game.

Setup: The challenger runs the **Set-Up** algorithm of the HIBE and gives the public parameters to $\mathcal{A}$.

Phase 1: $\mathcal{A}$ makes a number of key extraction queries adaptively. For a query for an identity vector $\overrightarrow{\mathsf{id}}$, the challenger responds with a key $\mathcal{SK}_{\overrightarrow{\mathsf{id}}}$.

Challenge: $\mathcal{A}$ provides a challenge identity $\overrightarrow{\mathsf{id}}^*$ and two equal length messages M_0 and M_1 to the challenger with the restriction that $\overrightarrow{\mathsf{id}}^*$ should not have been queried in **Phase 1**. The challenger then chooses a bit β uniformly at random from $\{0, 1\}$ and returns an encryption of M_β for $\overrightarrow{\mathsf{id}}^*$ to $\mathcal{A}$.

Phase 2: $\mathcal{A}$ issues more key extraction queries as in **Phase 1** with the restriction that no identity $\overrightarrow{\mathsf{id}}$ is a prefix of $\overrightarrow{\mathsf{id}}^*$.

Guess: $\mathcal{A}$ outputs a bit β'.

If $\beta = \beta'$, then $\mathcal{A}$ wins the game. The advantage of $\mathcal{A}$ in breaking the security of the HIBE scheme is given by

$$\mathsf{Adv}^{\mathcal{A}}_{\mathrm{HIBE}} = \left| \Pr[\beta = \beta'] - \frac{1}{2} \right|.$$

The HIBE scheme is secure if all probabilistic adversaries running in time bounded by a polynomial in κ have negligible (in κ) advantage in winning the above game.

A.5 Lewko-Waters IBE

Here we provide the construction of the IBE in the Type-3 setting given by Lewko and Waters [16]. The description in [16] consists of the usual ciphertexts and keys as well as the so-called semi-functional ciphertexts and keys. The semi-functional versions are obtained by modifying the normal versions. We use a compact notation to denote normal and semi-functional ciphertexts and keys. The group elements shown in curly brackets $\{\}$ are the semi-functional components. To get the scheme itself, these components should be ignored.

Let P_1 and P_2 be random generators of $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively. Let $Q_1, U_1 \in_R \mathbb{G}_1$ and $Q_2, U_2 \in \mathbb{G}_2$ be such that $Q_2 \sim Q_1$ and $U_2 \sim U_1$. Choose F_2 at random from $\mathbb{G}_2$, a, v, v' from $\mathbb{Z}_p$ and define $V_2 = vF_2, V_2' = v'F_2$. Define $\tau = v + av'$ so that $\tau F_2 = V_2 + aV_2'$. Identities are elements of $\mathbb{Z}_p$. The public parameters and master secret are given by

$\mathcal{PP}$: $(P_1, aP_1, \tau P_1, Q_1, aQ_1, \tau Q_1, U_1, aU_1, \tau U_1, e(P_1, P_2)^\alpha)$
$\mathcal{MSK}$: $(\alpha P_2, P_2, V_2, V_2', Q_2, U_2, F_2)$.

The randomisers for the ciphertext and key are s and t, r_1, r_2 respectively. These are elements of $\mathbb{Z}_p$. For the semi-functional components, μ, π_C and γ, π_K are chosen at random from $\mathbb{Z}_p$. Also elements $V_1', F_1 \in \mathbb{G}_1$ are such that $V_1' \sim V_2'$ and $F_1 \sim F_2$.

Ciphertext:
$C_0 = M \times e(P_1, P_2)^{\alpha s}$
$C_{1,1} = s(\mathsf{id}Q_1 + U_1)$, $C_{1,2} = as(\mathsf{id}Q_1 + U_1)\{+\mu\pi_C F_1\}$, $C_{1,3} = -\tau s(\mathsf{id}Q_1 + U_1)\{-\mu\pi_C V_1'\}$
$C_{2,1} = sP_1$, $C_{2,2} = asP_1\{+\mu F_1\}$, $C_{2,3} = -\tau sP_1\{-\mu V_1'\}$

Key:
$K_{1,1} = tP_2 + r_1V_2\{-a\gamma F_2\}$, $K_{1,2} = r_1V_2'\{+\gamma F_2\}$, $K_{1,3} = r_1F_2$
$K_{2,1} = \alpha P_2 + t(\mathsf{id}Q_2 + U_2) + r_1V_2\{-a\gamma\pi_K F_2\}$, $K_{2,2} = r_2V_2'\{+\gamma\pi_K F_2\}$, $K_{2,3} = r_2F_2$