Identity-based Hierarchical Key-insulated Encryption without Random Oracles*

Yohei Watanabe[1,2,†] and Junji Shikata[3,4]

[1] Graduate School of Informatics and Engineering,
The University of Electro-Communications, Tokyo, Japan

[2] Information Technology Research Institute (ITRI),
National Institute of Advanced Industrial Science and Technology (AIST), Tokyo, Japan

[3] Graduate School of Environment and Information Sciences,
Yokohama National University, Yokohama, Japan

[4] Institute of Advanced Sciences,
Yokohama National University, Yokohama, Japan

watanabe@uec.ac.jp, shikata@ynu.ac.jp

January 5, 2017

Abstract

Key-insulated encryption is one of the effective solutions to a key exposure problem. At Asiacrypt'05, Hanaoka et al. proposed an identity-based hierarchical key-insulated encryption (hierarchical IKE) scheme. Although their scheme is secure in the random oracle model, it has a "hierarchical key-updating structure," which is attractive functionality that enhances key exposure resistance.

In this paper, we first propose the hierarchical IKE scheme without random oracles. Our hierarchical IKE scheme is secure under the symmetric external Diffie–Hellman (SXDH) assumption, which is known as the simple and static one. Particularly, in the non-hierarchical case, our construction is the first IKE scheme that achieves constant-size parameters including public parameters, secret keys, and ciphertexts.

Furthermore, we also propose the first public-key-based key-insulated encryption (PK-KIE) in the hierarchical setting by using our technique.

Keywords: Key-insulated encryption, identity-based hierarchical key-insulated encryption, hierarchical identity-based encryption, asymmetric pairing.

1 Introduction

1.1 Background

A key exposure problem is unavoidable since human errors cannot seem to be eliminated in the future, and many researchers have tackled this problem in the modern cryptography area thus far. *Key-insulated encryption*, which is introduced by Dodis et al. [14], is one of the effective solutions to the key exposure problem. Specifically, they proposed public-key encryption (PKE) with the key-insulated property, which is called *public-key-based key-insulated encryption* (PK-KIE). In PK-KIE, a receiver has two kinds of secret keys, so-called *a decryption key* and *a helper key*. The decryption key is a short-term key for decrypting ciphertexts, and is periodically updated by the helper key. More specifically, the lifetime of the system is

*A preliminary version of this paper appears in PKC 2016 [31]. This is the full version.

†The author is supported by JSPS Research Fellowships for Young Scientists. Part of this work was done while the author was a Ph.D student at Graduate School of Environment and Information Sciences, Yokohama National University.

divided into discrete time periods, and the receiver can decrypt the ciphertext encrypted at some time period t by using a decryption key dk_t updated by the helper key at the same time period t. The decryption key and the helper key are stored in a powerful but insecure device such as laptops and smartphones and in a physically-secure but computationally-limited device such as USB pen drives, respectively. Traditionally, in key-insulated cryptography, the following two kinds of security notions are considered:

(a) If a number of decryption keys $\{dk_{t_1}, dk_{t_2}, \ldots, dk_{t_q}\}$ are exposed, no information on plaintexts encrypted at other time periods is leaked.

(b) Even if a helper key is exposed, the security is not compromised unless at least one decryption key is exposed.

A key-insulated cryptosystem is said to be secure if it satisfies (a); and it is *strongly secure* if it satisfies both (a) and (b). As seen above, key-insulated encryption can significantly reduce the impact of the exposure.

Following a seminal work by Dodis et al. [14], many cryptographers have proposed several kinds of key-insulated cryptographic schemes such as symmetric-key-based key-insulated encryption [16], key-insulated signatures [15], and parallel key-insulated encryption [19, 20, 25]. In addition to key-insulated cryptography, researchers have tackled the key exposure problem in various flavors. In forward-secure cryptography [1, 7], users update their own secret keys at the beginning of each time period. Forward security requires that an adversary cannot get any information on plaintexts encrypted at previous time periods even if the secret key for current time period is exposed. Intrusion-resilient cryptography [12, 13, 22] realizes both key-insulated security and forward security simultaneously at the sacrifice of efficiency and practicality.

In this paper, we focus on the key-insulation paradigm in the identity-based setting. Identity-based encryption (IBE) has been widely studied thus far, and therefore we believe that the identity-based key-insulated security has a huge influence on the research on IBE and its applications. Also, developing key-insulated cryptography in the identity-based area is the first step to consider the key-insulated security in attribute-based encryption [3, 28] and functional encryption [6], which are expected to be used in cloud environments. However, in the IBE context, there are only few researches on key-insulation. Hanaoka et al. [21] introduced *a hierarchical key-updating mechanism*, and proposed an IBE scheme with hierarchical key-insulation, which is called an identity-based hierarchical key-insulated encryption (hierarchical IKE for short) scheme, in the random oracle model. In their hierarchy, helper keys are assigned to each level, and decryption keys are assigned to the lowest level. Not only decryption keys but also helper keys can be updated by a higher-level helper key. Since this "hierarchy" is not the same as that of hierarchical IBE (HIBE) [18], only applying techniques used in the HIBE context is insufficient for constructing secure (in particular, *strongly* secure) IKE schemes. The hierarchical property is attractive since it enhances resistance to key exposure and there seem to be various applications due to progress in information technology (e.g., the popularization of smartphones). Let us consider an example of 3-level hierarchical key-insulation: Suppose that each employee has a business smartphone, a laptop, and a PC installed at his office. A decryption key is stored in the smartphone, and it is updated by a 1-st level helper key stored in his laptop every day. However, the 1-st level helper key might be leaked since he carries around with the laptop, and connects to the Internet via the laptop. Thus, the 1-st level helper key is also updated by a 2-nd level helper key stored in his PC every two–three weeks. Since the PC is not completely isolated from the Internet, his boss updates the 2-nd level helper key by a 3-rd level helper key stored in an isolated private device every two–three months. Thus, we believe hierarchical IKE has many potential applications.

After the proposal of hierarchical IKE by Hanaoka et al., two (non-hierarchical) IKE schemes with additional properties in the standard model were proposed. One is the so-called *parallel* IKE scheme, which was proposed by Weng et al. [34]. The other is the so-called *threshold* IKE scheme, which was proposed by Weng et al. [35]. These two schemes enhance the resistance to helper key exposure by splitting a helper key into multiple ones. However, once the (divided) helper key is leaked, the security cannot be recovered. Again, we emphasize that the hierarchical key-insulated structure is useful since even if some helper key is exposed, it can be updated. However, there have been no hierarchical IKE schemes without random oracles thus far.

1.2 Our Contribution

In this paper, we propose an IBE scheme with ℓ-level hierarchical key-insulation, which is called an ℓ-level hierarchical IKE scheme, such that: (1) Security is proved under simple computational assumptions in the standard model; and (2) all parameters including public parameters, secret keys, and ciphertexts achieve constant size in the non-hierarchical case (i.e., $\ell = 1$).

Specifically, The proposed ℓ-level hierarchical IKE scheme is strongly secure against chosen plaintext attacks (CPA-secure) under the symmetric external Diffie-Hellman (SXDH) assumption, which is a static and simple one. Our (hierarchical) IKE scheme is based on the Jutla-Roy (H)IBE [24] and its variant [27]. Further, the proposed scheme achieves the constant-size parameters when $\ell = 1$, whereas public parameters of the (not hierarchical) existing scheme [35] depend on sizes of identity spaces (also see Section 4.1 for comparison). We can also realize an ℓ-level hierarchical IKE scheme strongly secure against chosen ciphertext attacks (CCA-secure) based on an well-known transformation [5]. Furthermore, we can extend our technique to the public-key setting. Namely, we formalize public-key encryption with hierarchical key insulation (hierarchical PK-KIE for short), and propose a concrete construction of a CCA-secure hierarchical PK-KIE scheme.

In the following, we explain why a naive solution is insufficient and why achieving (1) and (2) is challenging.

Why a (trivial) hierarchical IKE scheme from HIBE is insufficient.[1] One may think that a hierarchical IKE scheme can be easily obtained from an arbitrary HIBE scheme. However, the resulting IKE scheme is insecure in our security model, which was first formalized in [21], since our security model captures the *strong* security notion. More specifically, a trivial construction is as follows. Let $sk_{\mathtt{I}}$ be a secret key for some identity $\mathtt{I}$ in HIBE, and $hk_{\mathtt{I}}^{(\ell)}$ be an ℓ-th level helper key for $\mathtt{I}$ in ℓ-level hierarchical IKE. We set $sk_{\mathtt{I}}$ as $hk_{\mathtt{I}}^{(\ell)}$, and lower-level helper keys and decryption keys can be obtained from $sk_{\mathtt{I}}$ by regarding time periods as descendants' identity. However, it is easy to see that if the ℓ-th level helper key (i.e., $sk_{\mathtt{I}}$) is exposed, then an adversary can obtain all lower-level keys, and thus the resulting scheme does not meet the strong security. In fact, Bellare and Palacio [2] showed that secure (*not strongly secure*) PK-KIE is equivalent to IBE for a similar reason.

Difficulties in constructing a constant-size IKE scheme from simple computational assumptions. The main difficulty in constructing an IKE scheme is that an adversary can get various keys for a target identity $\mathtt{I}^*$, whereas the adversary cannot get a secret key for $\mathtt{I}^*$ in (H)IBE. This point makes a construction methodology non-trivial. In fact, it seems difficult to apply the Waters dual-system IBE [33] (and its variant [26]) as the underlying IBE scheme of IKE schemes as follows. Technically, in their scheme each of secret keys and ciphertexts contains some random element, so-called tag_K and tag_C, respectively. In the dual system encryption methodology, the challenge ciphertext and secret keys gradually turn into semi-functional forms. The tags are used in transition from G_{k-1} to G_k, where G_k denotes a security game that the first k secret keys issued to a secret-key extraction oracle are semi-functional. In the transition, some pairwise independent function is embedded into public parameters in advance to cancel inconvenient values to simulate the games. The tag tag_K of a secret key for k-th identity $\mathtt{I}_k$ issued to the oracle and the tag tag_C of the challenge ciphertext for the target identity $\mathtt{I}^*$ are generated by inputting $\mathtt{I}$ into the pairwise independent function, respectively. Although it holds $tag_K = tag_C$ if $\mathtt{I}_k = \mathtt{I}^*$, the proof works well since it is enough to generate only tag_K for all identities $\mathtt{I} \neq \mathtt{I}^*$ and only tag_C for $\mathtt{I}^*$. However, in the IKE setting, not only tag_C but also tag_K for $\mathtt{I}^*$ have to be generated since an adversary can get leaked decryption keys and helper keys for $\mathtt{I}^*$, and hence, the proof does not go well. To overcome this challenging point, we set (the variant of) the Jutla-Roy IBE [24, 27], which is another type of constant-size IBE schemes, as the underlying scheme of our IKE scheme, and thus we can realize the first constant-size IKE scheme under the SXDH assumption. Further, we can also obtain the hierarchical IKE scheme by extending the technique into the hierarchical setting.

Refinement and improvement from the proceedings version [31]. We modify our main construction due to a security flaw of the previous construction in the proceeding version. Specifically, the modified construction provides a correct simulation of a *KI* oracle, which is an oracle that captures key exposure,

[1] This fact was also mentioned in [21].

in transition from Game_{k-1} to Game_k, where Game_k denotes a security game in which keys for the first k identities issued to oracles are semi-functional (for details, see the simulation of the *KI* oracle in Lemma 2, which shows the transition). Moreover, we change the statement of Lemma 3, which is the final transition in the security proof, to make a reduction clearer. More specifically, we make a reduction to a computational problem in the lemma, whereas we made information-theoretic reduction in the proceedings version.

Further, we newly propose hierarchical PK-KIE, which did not appear in the proceedings version, by extending our technique.

1.3 Paper Organization

In Section 2, we describe the notation used in this paper, asymmetric pairings, complexity assumptions, and functions which map time to discrete time periods. In Section 3, we give a model and security definition of hierarchical IKE. In Section 4, we propose a direct construction of our hierarchical IKE scheme, and give the efficiency comparison among our scheme and existing schemes. In Section 5, we show the security proof of our scheme. In Section 6, we show a CCA-secure hierarchical IKE scheme. In Section 7, we formalize and propose a hierarchical PK-KIE scheme. In Section 8, we conclude this paper.

2 Preliminaries

Notation. In this paper, "probabilistic polynomial-time" is abbreviated as "PPT". For a prime p, let $\mathbb{Z}_p := \{0, 1, \ldots, p-1\}$ and $\mathbb{Z}_p^\times := \mathbb{Z}_p \setminus \{0\}$. If we write $(y_1, y_2, \ldots, y_m) \leftarrow \mathcal{A}(x_1, x_2, \ldots, x_n)$ for an algorithm $\mathcal{A}$ having n inputs and m outputs, it means to input $x_1, x_2, \ldots, x_n$ into $\mathcal{A}$ and to get the resulting output $y_1, y_2, \ldots, y_m$. We write $(y_1, y_2, \ldots, y_m) \leftarrow \mathcal{A}^{\mathcal{O}}(x_1, x_2, \ldots, x_n)$ to indicate that an algorithm $\mathcal{A}$ that is allowed to access an oracle $\mathcal{O}$ takes $x_1, x_2, \ldots, x_n$ as input and outputs $(y_1, y_2, \ldots, y_m)$. If $\mathcal{X}$ is a set, we write $x \xleftarrow{\$} \mathcal{X}$ to mean the operation of picking an element x of $\mathcal{X}$ uniformly at random. We use λ as a security parameter. $\mathcal{M}$ and $\mathcal{I}$ denote sets of plaintexts and IDs, respectively, which are determined by a security parameter λ.

Bilinear Group. A bilinear group generator $\mathcal{G}$ is an algorithm that takes a security parameter λ as input and outputs a bilinear group $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$, where p is a prime, $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ are multiplicative cyclic groups of order p, g_1 and g_2 are (random) generators of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively, and e is an efficiently computable and non-degenerate bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ with the following bilinear property: For any $u, u' \in \mathbb{G}_1$ and $v, v' \in \mathbb{G}_2$, $e(uu', v) = e(u, v)e(u', v)$ and $e(u, vv') = e(u, v)e(u, v')$.

A bilinear map e is called symmetric or a "Type-1" pairing if $\mathbb{G}_1 = \mathbb{G}_2$. Otherwise, it is called asymmetric. In the asymmetric setting, e is called a "Type-2" pairing if there is an efficiently computable isomorphism either from $\mathbb{G}_1$ to $\mathbb{G}_2$ or from $\mathbb{G}_2$ to $\mathbb{G}_1$. If no efficiently computable isomorphisms are known, then it is called a "Type-3" pairing. In this paper, we focus on the Type-3 pairing, which is the most efficient setting in terms of group sizes (of $\mathbb{G}_1$) and operations. For details, see [9, 17]. **Symmetric External Diffie–Hellman (SXDH) Assumption.** We give the definition of the decisional Diffie–Hellman (DDH) assumption in $\mathbb{G}_1$ and $\mathbb{G}_2$, which are called the DDH1 and DDH2 assumptions, respectively.

Let $\mathcal{A}$ be a PPT adversary and we consider $\mathcal{A}$'s advantage against the DDHi problem ($i = 1, 2$) as follows.

$$Adv^{DDHi}_{\mathcal{G},\mathcal{A}}(\lambda) := \left| \Pr\left[b' = b \;\middle|\; \begin{array}{l} D := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e) \leftarrow \mathcal{G}, \\ c_1, c_2 \xleftarrow{\$} \mathbb{Z}_p,\ b \xleftarrow{\$} \{0,1\}, \\ \text{if } b = 0 \text{ then } T := g_i^{c_1 c_2}, \\ \text{else } T \xleftarrow{\$} \mathbb{G}_i, \\ b' \leftarrow \mathcal{A}(\lambda, D, g_1, g_2, g_i^{c_1}, g_i^{c_2}, T) \end{array} \right] - \frac{1}{2} \right|.$$

Definition 1 (DDHi Assumption)**.** *The DDHi assumption relative to a generator $\mathcal{G}$ holds if for all PPT adversaries $\mathcal{A}$, $Adv^{DDHi}_{\mathcal{G},\mathcal{A}}(\lambda)$ is negligible in λ.*

Definition 2 (SXDH Assumption)**.** *We say that the SXDH assumption relative to a generator $\mathcal{G}$ holds if both the DDH1 and DDH2 assumptions relative to $\mathcal{G}$ hold.*

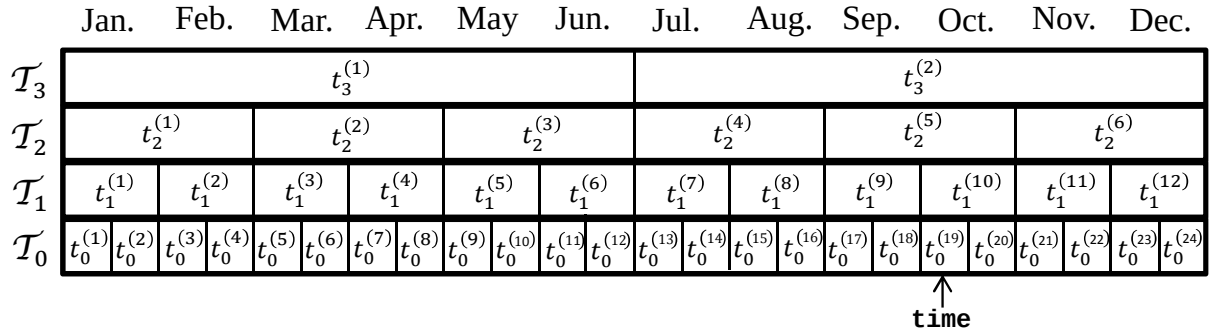

Figure 1: Intuition of time-period map functions.

Time-period Map Functions. In this paper, we deal with *several kinds of time periods* since we consider that update intervals of each level key are different. For example, in some practical applications, it might be suitable that a decryption key (i.e. 0-th level key) and a 1-st level helper key should be updated every day and every month, respectively. To describe such different update intervals of each level key, we use a certain functions, which is so-called *time-period map functions*. This functions were also used in [21]. Now, let $\mathcal{T}$ be a (possibly countably infinite) set of *time*, and $\mathcal{T}_j$ $(0 \leq j \leq \ell - 1)$ be a finite set of *time periods*. We assume $|\mathcal{T}_0| \geq |\mathcal{T}_1| \geq \cdots \geq |\mathcal{T}_{\ell-1}|$. This means that a lower-level key is updated more frequently than the higher-level keys. Then, we assume there exists a function T_j $(0 \leq j \leq \ell - 1)$ which map time $\mathtt{time} \in \mathcal{T}$ to a time period $t_j \in \mathcal{T}_j$. For the understanding of readers, by letting $\mathtt{time} = \mathtt{9:59/7th/Oct./2016}$ and $\ell := 4$, we give an example in Figure 1 and below. For example, we have $T_0(\mathtt{time}) = t_0^{(19)} = \mathtt{1st\text{-}15th/Oct./2016}$, $T_1(\mathtt{time}) = t_1^{(10)} = \mathtt{Oct./2016}$, $T_2(\mathtt{time}) = t_2^{(5)} = \mathtt{Sep.\text{-}Oct./2016}$, and $T_3(\mathtt{time}) = t_3^{(2)} = \mathtt{Jul.\text{-}Dec./2016}$. Namely, in this example, it is assumed that the decryption key, and 1-st, 2-nd, and 3-rd helper keys are updated every half a month, every month, every two months, and every half a year. Further, we can also define a function T_ℓ such that $T_\ell(\mathtt{time}) = 0$ for all $\mathtt{time} \in \mathcal{T}$.

3 Identity-based Hierarchical Key-insulated Encryption

3.1 The Model

In an ℓ-level hierarchical IKE, a key generation center (KGC) generates an initial decryption key $dk_{\mathtt{I},0}$ and ℓ initial helper keys $hk_{\mathtt{I},0}^{(1)}, hk_{\mathtt{I},0}^{(2)}, \ldots, hk_{\mathtt{I},0}^{(\ell)}$ as a secret key for a user $\mathtt{I}$. Suppose that all time-period map functions $T_0, T_1, \ldots, T_{\ell-1}$ are available to all users. The key-updating procedure when the user wants to get a decryption key at current time $\mathtt{time} \in \mathcal{T}$ from the initial helper keys is as follows. The ℓ-th level helper key $hk_{\mathtt{I},0}^{(\ell)}$ is a long-term one and is never updated. First, the user generates *key update* $\delta_{t_{\ell-1}}^{(\ell-1)}$ for the $(\ell-1)$-th level helper key from $hk_{\mathtt{I},0}^{(\ell)}$ and a time period $t_{\ell-1} := T_{\ell-1}(\mathtt{time}) \in \mathcal{T}_{\ell-1}$. Then, the $(\ell-1)$-th level helper key $hk_{\mathtt{I},0}^{(\ell-1)}$ can be updated by the key update $\delta_{t_{\ell-1}}^{(\ell-1)}$, and the user get the helper key $hk_{\mathtt{I},t_{\ell-1}}^{(\ell-1)}$ at the time period $t_{\ell-1}$. Similarly, the i-th level helper key $hk_{\mathtt{I},t_i}^{(i)}$ at the time period $t_i := T_i(\mathtt{time}) \in \mathcal{T}_i$ can be obtained from $hk_{\mathtt{I},0}^{(i)}$ and $\delta_{t_i}^{(i)}$, where $\delta_{t_i}^{(i)}$ is generated from the $(i+1)$-th level helper key $hk_{\mathtt{I},t_{i+1}}^{(i+1)}$. The user can finally get the decryption key $dk_{\mathtt{I},t_0}$ at a time period $t_0 := T_0(\mathtt{time}) \in \mathcal{T}_0$ from the 1-st level helper key $hk_{\mathtt{I},T_1(\mathtt{time})}^{(1)}$. Anyone can encrypt a plaintext M with the identity $\mathtt{I}$ and current time $\mathtt{time}^*$, and the user can decrypt the ciphertext C with his decryption key $dk_{\mathtt{I},t_0}$ if and only if $t_0 = T_0(\mathtt{time}^*)$. At $\mathtt{time}' \in \mathcal{T}$, the user can update the time period of the decryption key from any time period t_0 to $t_0' := T_0(\mathtt{time}') \in \mathcal{T}_0$ by using key update $\delta_{T_0(\mathtt{time}')}^{(0)}$. The key update $\delta_{T_0(\mathtt{time}')}^{(0)}$ can be obtained from $hk_{\mathtt{I},t_1'}^{(1)}$ if and only if $t_1' = T_1(\mathtt{time}')$. If not, it is necessary to get $\delta_{T_1(\mathtt{time}')}^{(1)}$ and update $hk_{\mathtt{I},t_1'}^{(1)}$. In this manner, the decryption and helper keys are updated.

An ℓ-level hierarchical IKE scheme Π_{IKE} consists of six-tuple algorithms (PGen, Gen, Δ-Gen, Upd, Enc, Dec) defined as follows. For simplicity, we omit a public parameter in the input of all algorithms except for

the PGen algorithm.

- $(pp, mk) \leftarrow \mathsf{PGen}(\lambda, \ell)$: A probabilistic algorithm for parameter generation. It takes a security parameter λ and the maximum hierarchy depth ℓ as input, and outputs a public parameter pp and a master key mk.
- $(dk_{\mathtt{I},0}, hk_{\mathtt{I},0}^{(1)}, \ldots, hk_{\mathtt{I},0}^{(\ell)}) \leftarrow \mathsf{Gen}(mk, \mathtt{I})$: An algorithm for user key generation. It takes mk and an identity $\mathtt{I} \in \mathcal{I}$ as input, and outputs an initial secret key $dk_{\mathtt{I},0}$ associated with $\mathtt{I}$ and initial helper keys $hk_{\mathtt{I},0}^{(1)}, \ldots, hk_{\mathtt{I},0}^{(\ell)}$, where $hk_{\mathtt{I},0}^{(i)}$ $(1 \leq i \leq \ell)$ is assumed to be stored user's i-th level private device.
- $\delta_{T_{i-1}(\mathtt{time})}^{(i-1)}$ or $\perp \leftarrow \Delta\text{-}\mathsf{Gen}(hk_{\mathtt{I},t_i}^{(i)}, \mathtt{time})$: An algorithm for key update generation. It takes an i-th helper key $hk_{\mathtt{I},t_i}^{(i)}$ at a time period $t_i \in \mathcal{T}_i$ and current time $\mathtt{time}$ as input, and outputs key update $\delta_{T_{i-1}(\mathtt{time})}^{(i-1)}$ if $t_i = T_i(\mathtt{time})$; otherwise, it outputs $\perp$.
- $hk_{\mathtt{I},\tau_i}^{(i)} \leftarrow \mathsf{Upd}(hk_{\mathtt{I},t_i}^{(i)}, \delta_{\tau_i}^{(i)})$: A probabilistic algorithm for decryption key generation. It takes an i-th helper key $hk_{\mathtt{I},t_i}^{(i)}$ at a time period $t_i \in \mathcal{T}_i$ and key update $\delta_{\tau_i}^{(i)}$ at a time period $\tau \in \mathcal{T}_i$ as input, and outputs a renewal i-th helper key $hk_{\mathtt{I},\tau_i}^{(i)}$ at τ. Note that for any $t_0 \in \mathcal{T}_0$, $hk_{\mathtt{I},t_0}^{(0)}$ means $dk_{\mathtt{I},t_0}$.
- $\langle C, \mathtt{time} \rangle \leftarrow \mathsf{Enc}(\mathtt{I}, \mathtt{time}, M)$: A probabilistic algorithm for encryption. It takes an identity $\mathtt{I}$, current time $\mathtt{time}$, and a plaintext $M \in \mathcal{M}$ as input, and outputs a pair of a ciphertext and current time $\langle C, \mathtt{time} \rangle$.
- M or $\perp \leftarrow \mathsf{Dec}(dk_{\mathtt{I},t_0}, \langle C, \mathtt{time} \rangle)$: A deterministic algorithm for decryption. It takes $dk_{\mathtt{I},t_0}$ and $\langle C, \mathtt{time} \rangle$ as input, and outputs M or $\perp$, where $\perp$ indicates decryption failure.

In the above model, we assume that Π_{IKE} meets the following correctness property: For all security parameter λ, all $\ell := poly(\lambda)$, all $(mk, pp) \leftarrow \mathsf{PGen}(\lambda, \ell)$, all $M \in \mathcal{M}$, all $(dk_{\mathtt{I},0}, hk_{\mathtt{I},0}^{(1)}, \ldots, hk_{\mathtt{I},0}^{(\ell)}) \leftarrow \mathsf{Gen}(mk, \mathtt{I})$, and all $\mathtt{time} \in \mathcal{T}$, it holds that $M \leftarrow \mathsf{Dec}(dk_{\mathtt{I},T_0(\mathtt{time})}, \mathsf{Enc}(\mathtt{I}, \mathtt{time}, M))$, where $dk_{\mathtt{I},T_0(\mathtt{time})}$ is generated as follows: For $i = \ell, \ldots, 1$, $hk_{\mathtt{I},T_{i-1}(\mathtt{time})}^{(i-1)} \leftarrow \mathsf{Upd}(hk_{\mathtt{I},t_{i-1}}^{(i-1)}, \Delta\text{-}\mathsf{Gen}(hk_{\mathtt{I},T_i(\mathtt{time})}^{(i)}, \mathtt{time}))$, where some $t_i \in \mathcal{T}_i$ and $hk_{\mathtt{I},T_0(\mathtt{time})}^{(0)} := dk_{\mathtt{I},T_0(\mathtt{time})}$.

3.2 Security Definition

We consider a security notion for indistinguishability against key exposure and chosen plaintext attack for IKE (IND-KE-CPA). Let $\mathcal{A}$ be a PPT adversary, and $\mathcal{A}$'s advantage against IND-KE-CPA security is defined by

$$Adv_{\Pi_{IKE},\mathcal{A}}^{IND\text{-}KE\text{-}CPA}(\lambda, \ell) := \left| \Pr \left[b' = b \;\middle|\; \begin{array}{l} (pp, mk) \leftarrow \mathsf{PGen}(\lambda, \ell), \\ (M_0^*, M_1^*, \mathtt{I}^*, \mathtt{time}^*, state) \leftarrow \mathcal{A}^{KG(\cdot), KI(\cdot,\cdot,\cdot)}(\mathsf{find}, pp), \\ b \xleftarrow{\$} \{0,1\}, C^* \leftarrow \mathsf{Enc}(\mathtt{I}^*, \mathtt{time}^*, M_b^*), \\ b' \leftarrow \mathcal{A}^{KG(\cdot), KI(\cdot,\cdot,\cdot)}(\mathsf{guess}, C^*, state) \end{array} \right] - \frac{1}{2} \right|.$$

where $KG(\cdot)$ and $KI(\cdot,\cdot,\cdot)$ are defined as follows.

$\boldsymbol{KG}(\cdot)$**:** For a query $\mathtt{I} \in \mathcal{I}$, it stores and returns $(dk_{\mathtt{I},0}, hk_{\mathtt{I},0}^{(1)}, \ldots, hk_{\mathtt{I},0}^{(\ell)})$ by running $\mathsf{Gen}(mk, \mathtt{I})$.

$\boldsymbol{KI}(\cdot,\cdot,\cdot)$**:** For a query $(i, \mathtt{I}, \mathtt{time}) \in \{0, 1, \ldots, \ell\} \times \mathcal{I} \times \mathcal{T}$, it returns $hk_{\mathtt{I},T_i(\mathtt{time})}^{(i)}$ by running $\delta_{T_{j-1}(\mathtt{time})}^{(j-1)} \leftarrow \Delta\text{-}\mathsf{Gen}(hk_{\mathtt{I},T_j(\mathtt{time})}^{(j)}, \mathtt{time})$ and $hk_{\mathtt{I},T_{j-1}(\mathtt{time})}^{(j-1)} \leftarrow \mathsf{Upd}(hk_{\mathtt{I},t}^{(j-1)}, \delta_{T_{j-1}(\mathtt{time})}^{(j-1)})$ for $j = \ell, \ldots, i{+}1$ (if $(dk_{\mathtt{I},0}, hk_{\mathtt{I},0}^{(1)}$ $\ldots, hk_{\mathtt{I},0}^{(\ell)})$ is not stored, it first generates and stores them by running Gen).

$\mathtt{I}^*$ is never issued to the KG oracle. $\mathcal{A}$ can issue any queries $(i, \mathtt{I}, \mathtt{time})$ to the KI oracle if there exists at least one *special level* $j \in \{0, 1, \ldots, \ell\}$ such that

1. For any $\mathtt{time} \in \mathcal{T}$, $(j, \mathtt{I}^*, \mathtt{time})$ is never issued to KI.

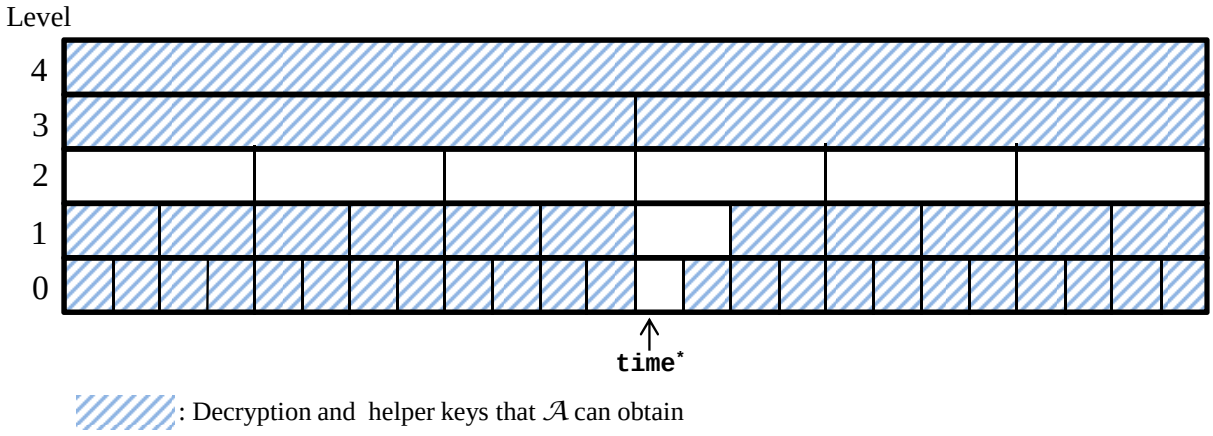

Figure 2: Pictorial representation of secret keys for $\mathtt{I}^*$ that $\mathcal{A}$ can obtain by issuing to *KI*.

2. For any $(i, \mathtt{time}) \in \{0, 1, \ldots, j-1\} \times \mathcal{T}$ such that $T_i(\mathtt{time}) = T_i(\mathtt{time}^*)$, $(i, \mathtt{I}^*, \mathtt{time})$ is never issued to *KI*.

In Figure 2, we give intuition of keys that $\mathcal{A}$ can obtain by issuing to the *KI* oracle. In this example, let $\ell = 4$ and a special level $j = 2$.

Definition 3 (IND-KE-CPA [21])**.** *An ℓ-level hierarchical IKE scheme Π_{IKE} is said to be IND-KE-CPA secure if for all PPT adversaries $\mathcal{A}$, $Adv^{IND\text{-}KE\text{-}CPA}_{\Pi_{IKE},\mathcal{A}}(\lambda, \ell)$ is negligible in λ.*

Remark 1. *As also noted in [21], there is no need to consider key update exposure explicitly (i.e. no need to consider an oracle which returns any key update as much as possible) since in the above definition, $\mathcal{A}$ can get such key update from helper keys obtained from the KI oracle.*

Remark 2. *As explained in Section 1, in key-insulated cryptography including the public key setting [2, 14, 19] and the identity-based setting [21, 34, 35], two kinds of security notions have been traditionally considered: standard security and strong security. In most of previous works [2, 14, 19, 20, 21, 25, 34, 35], authors have considered how their scheme could achieve the strong security. We note that IND-KE-CPA security actually includes the strong security, and the fact is easily checked by setting $\ell = 1$.*

By modifying the above IND-KE-CPA game so that $\mathcal{A}$ can access to the decryption oracle $Dec(\cdot, \cdot)$, which receives $(\mathtt{I}, \langle C, \mathtt{time}\rangle)$ and returns M or $\perp$, we can also define indistinguishability against key exposure and chosen ciphertext attack for IKE (IND-KE-CCA). $\mathcal{A}$ is not allowed to issue $(\mathtt{I}^*, \langle C^*, \mathtt{time}\rangle)$ such that $T_0(\mathtt{time}) = T_0(\mathtt{time}^*)$ to *Dec*. Let $Adv^{IND\text{-}KE\text{-}CCA}_{\Pi_{IKE},\mathcal{A}}(\lambda, \ell)$ be $\mathcal{A}$'s advantage against IND-KE-CCA security.

Definition 4 (IND-KE-CCA [21])**.** *An ℓ-level hierarchical IKE scheme Π_{IKE} is said to be IND-KE-CCA secure if for all PPT adversaries $\mathcal{A}$, $Adv^{IND\text{-}KE\text{-}CCA}_{\Pi_{IKE},\mathcal{A}}(\lambda, \ell)$ is negligible in λ.*

4 Our Construction

Our basic idea is a combination of (the variant of) the Jutla-Roy HIBE [24, 27] and threshold secret sharing schemes [4, 29]. We prepare two secrets $B^{(x)}$ and $B^{(y)}$. Each secret $B^{(j)}$ $(j \in \{x, y\})$ is divided into ℓ shares $\beta_0^{(j)}, \ldots, \beta_{\ell-1}^{(j)}$, and both the secrets and shares are used in exponent of a generator $g_2 \in \mathbb{G}_2$. $B^{(x)}$ and $B^{(y)}$ are embedded into the exponent of a (first-level) secret key for $\mathtt{I}$ of the Jutla-Roy HIBE, and the resulting key is used as an ℓ-th level initial helper key $hk^{(\ell)}_{\mathtt{I},0}$. Roughly speaking, $B^{(x)}$ and $B^{(y)}$ work as "noises". Other initial helper keys $hk^{(i)}_{\mathtt{I},0}$ $(1^l ei^l e^e ll - 1)$ and an initial decryption key $dk_{\mathtt{I},0}$ contain $(g_2^{-\beta_i^{(x)}}, g_2^{-\beta_i^{(y)}})$ and $(g_2^{-\beta_0^{(x)}}, g_2^{-\beta_0^{(y)}})$, respectively. As keys are generated for lower levels, shares are eliminated from the noises $B^{(x)}$ and $B^{(y)}$, respectively, and finally the noises are entirely removed when generating (or updating) a

decryption key. Intuitively, since there exists at lease one special level $j \in \{0, 1, \ldots, \ell\}$ in which any secret keys are never exposed, an adversary cannot get all shares $(\beta_i^{(x)}, \beta_i^{(y)})$. Hence, he cannot generate valid decryption keys that can decrypt the challenge ciphertext for $\mathtt{I}^*$ at $\mathtt{time}^*$.

An ℓ-level hierarchical IKE scheme Π_{IKE} =(PGen, Gen, Δ-Gen, Upd, Enc, Dec) is constructed as follows.

- $\mathsf{PGen}(\lambda, \ell)$: It runs $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, g_1, g_2, e) \leftarrow \mathcal{G}$, and chooses $x_0, y_0, \{(x_{1,j},\ y_{1,j})\}_{j=0}^{\ell}, x_2, y_2, x_3, y_3 \xleftarrow{\$} \mathbb{Z}_p$ and $\alpha \xleftarrow{\$} \mathbb{Z}_p^{\times}$, and sets

$$z = e(g_1, g_2)^{-x_0\alpha + y_0}, \quad u_{1,j} := g_1^{-x_{1,j}\alpha + y_{1,j}}\ (0 \le j \le \ell), \quad w_1 := g_1^{-x_2\alpha + y_2}, \quad h_1 := g_1^{-x_3\alpha + y_3}.$$

  It outputs

$$pp := (g_1, g_1^{\alpha}, \{u_{1,j}\}_{j=0}^{\ell}, w_1, h_1, g_2, \{(g_2^{x_{1,j}}, g_2^{y_{1,j}})\}_{j=0}^{\ell}, g_2^{x_2}, g_2^{x_3}, g_2^{y_2}, g_2^{y_3}, z),$$
$$mk := (x_0, y_0).$$

- $\mathsf{Gen}(mk, \mathtt{I})$: It chooses $\beta_0^{(x)}, \ldots, \beta_{\ell-1}^{(x)}, \beta_0^{(y)}, \ldots, \beta_{\ell-1}^{(y)}, r \xleftarrow{\$} \mathbb{Z}_p$, and let $B^{(j)} := \sum_{i=0}^{\ell-1} \beta_i^{(j)}$ for $j \in \{x, y\}$. It computes

$$R_j^{(y)} := g_2^{-\beta_j^{(y)}}\ (0 \le j \le \ell - 1), \quad R_j^{(x)} := g_2^{\beta_j^{(x)}}\ (0 \le j \le \ell - 1),$$
$$D_1 := (g_2^{y_2})^r, \quad D_1' := g_2^{y_0 + B^{(y)}} \left((g_2^{y_{1,\ell}})^{\mathtt{I}} g_2^{y_3}\right)^r,$$
$$D_2 := (g_2^{x_2})^{-r}, \quad D_2' := g_2^{-x_0 - B^{(x)}} \left((g_2^{x_{1,\ell}})^{\mathtt{I}} g_2^{x_3}\right)^{-r},$$
$$D_3 := g_2^r, \quad K_j := (g_2^{y_{1,j}})^r\ (0 \le j \le \ell - 1), \quad K_j' := (g_2^{x_{1,j}})^{-r}\ (0 \le j \le \ell - 1).$$

  It outputs

$$dk_{\mathtt{I},0} := (R_0^{(y)}, R_0^{(x)}),$$
$$hk_{\mathtt{I},0}^{(i)} := (R_i^{(y)}, R_i^{(x)})\ (1 \le i \le \ell - 1),$$
$$hk_{\mathtt{I},0}^{(\ell)} := (D_1, D_1', D_2, D_2', D_3, \{(K_j, K_j')\}_{j=0}^{\ell-1}).$$

- $\Delta\text{-}\mathsf{Gen}(hk_{\mathtt{I},t_i}^{(i)}, \mathtt{time})$: If $t_i \neq T_i(\mathtt{time})$, it outputs $\perp$. Otherwise, parse $hk_{\mathtt{I},t_i}^{(i)}$ as $(R_i^{(y)}, R_i^{(x)}, D_1, D_1', D_2, D_2', D_3, \{(K_j, K_j')\}_{j=0}^{i-1})$.[2] It chooses $\hat{r} \leftarrow \mathbb{Z}_p$, and let $t_j := T_j(\mathtt{time})$ $(i - 1 \le j \le \ell - 1)$. It computes

$$\hat{d}_1 := D_1 (g_2^{y_2})^{\hat{r}}, \quad \hat{d}_1' := D_1' (K_{i-1})^{t_{i-1}} \left((g_2^{y_{1,\ell}})^{\mathtt{I}} \prod_{j=i-1}^{\ell-1} \left((g_2^{y_{1,j}})^{t_j}\right) g_2^{y_3}\right)^{\hat{r}},$$
$$\hat{d}_2 := D_2 (g_2^{x_2})^{-\hat{r}}, \quad \hat{d}_2' := D_2' (K_{i-1}')^{t_{i-1}} \left((g_2^{x_{1,\ell}})^{\mathtt{I}} \prod_{j=i-1}^{\ell-1} \left((g_2^{x_{1,j}})^{t_j}\right) g_2^{x_3}\right)^{-\hat{r}},$$
$$\hat{d}_3 := D_3 g_2^{\hat{r}}, \quad \hat{k}_j := K_j (g_2^{y_{1,j}})^{\hat{r}}\ (0 \le j \le i - 2), \quad \hat{k}_j' := K_j' (g_2^{x_{1,j}})^{-\hat{r}}\ (0 \le j \le i - 2).$$

  It outputs $\delta_{t_{i-1}}^{(i-1)} := (\hat{d}_1, \hat{d}_1', \hat{d}_2, \hat{d}_2', \hat{d}_3, \{(\hat{k}_j, \hat{k}_j')\}_{j=0}^{i-2})$.[3]

- $\mathsf{Upd}(hk_{\mathtt{I},t_i}^{(i)}, \delta_{\tau_i}^{(i)})$: Parse $hk_{\mathtt{I},t_i}^{(i)}$ and $\delta_{\tau_i}^{(i)}$ as $(R_i^{(y)}, R_i^{(x)}, D_1, D_1', D_2, D_2', D_3, \{(K_j, K_j')\}_{j=0}^{i-1})$ and $(\hat{d}_1, \hat{d}_1', \hat{d}_2, \hat{d}_2', \hat{d}_3, \{(\hat{k}_j, \hat{k}_j')\}_{j=0}^{i-1})$, respectively. It outputs $hk_{\mathtt{I},\tau_i}^{(i)} := (\hat{R}_i^{(y)}, \hat{R}_i^{(x)}, \hat{D}_1, \hat{D}_1', \hat{D}_2, \hat{D}_2', \hat{D}_3, \{(\hat{K}_j, \hat{K}_j')\}_{j=0}^{i-1}) = (R_i^{(y)}, R_i^{(x)}, \hat{d}_1, \hat{d}_1' R_i^{(y)}, \hat{d}_2, \hat{d}_2' R_i^{(x)}, \hat{d}_3, \{\hat{k}_j, \hat{k}_j'\}_{j=0}^{i-1})$.

[2] In the case $i = \ell$, $R_i^{(y)}$ and $R_i^{(x)}$ mean empty strings, namely we have $hk_{\mathtt{I},0}^{(\ell)} := (D_1, D_1', D_2, D_2', D_3, \{(K_j, K_j')\}_{j=0}^{\ell-1})$.

[3] In the case $i = 1$, $\{(\hat{k}_j, \hat{k}_j')\}_{j=0}^{\ell-1}$ means an empty string, namely we have $\delta_{\mathtt{I},t_0}^{(0)} := (\hat{d}_1, \ldots, \hat{d}_5)$.

- $\mathsf{Enc}(\mathtt{I}, \mathtt{time}, M)$: It chooses $s, \mathtt{tag} \xleftarrow{\$} \mathbb{Z}_p$. For $M \in \mathbb{G}_T$, it computes

$$C_0 := Mz^s,\ C_1 := g_1^s,\ C_2 := (g_1^\alpha)^s,\ C_3 := \Big(\prod_{j=0}^{\ell-1}(u_{1,j}^{t_j})u_{1,\ell}^{\mathtt{I}}w_1^{\mathtt{tag}}h_1\Big)^s,$$

where $t_j := T_j(\mathtt{time})$ $(0 \le j \le \ell - 1)$. It outputs $C := (C_0, C_1, C_2, C_3, \mathtt{tag})$.

- $\mathsf{Dec}(dk_{\mathtt{I},t_0}, \langle C, \mathtt{time}\rangle)$: If $t_0 \neq T_0(\mathtt{time})$, then it outputs $\perp$. Otherwise, parse $dk_{\mathtt{I},t_0}$ and C as $(R_0^{(y)}, R_0^{(x)}, D_1, D_1', D_2, D_2', D_3)$ and $(C_0, C_1, C_2, C_3, \mathtt{tag})$, respectively. It computes

$$M = \frac{C_0 e(C_3, D_3)}{e(C_1, D_1^{\mathtt{tag}}D_1')e(C_2, D_2^{\mathtt{tag}}D_2')}.$$

We show the correctness of our Π_{IKE}. Suppose that r denotes internal randomness of $hk_{\mathtt{I},0}^{(\ell)}$, which are generated when running $\mathsf{Gen}(mk, \mathtt{I})$, and $r^{(j)}$ denotes internal randomness of $\delta_{\mathtt{I},t_{j-1}}^{(j-1)}$ $(1 \le j \le \ell)$, which is generated when running $\Delta\text{-}\mathsf{Gen}(hk_{\mathtt{I},t_j}^{(j)}, \mathtt{time})$. Then we can write $dk_{\mathtt{I},\tau_0} := (R_0^{(y)}, R_0^{(x)}, D_1, D_1', D_2,\ D_2', D_3)$ as

$$D_1 := g_2^{y_2\tilde{r}},\ D_1' := g_2^{y_0+\tilde{r}(\mathtt{I}y_{1,\ell}+\sum_{j=0}^{\ell-1}\left(t_j y_{1,j}\right)+y_3)},$$
$$D_2 := g_2^{x_2\tilde{r}},\ D_2' := g_2^{-x_0-\tilde{r}(\mathtt{I}x_{1,\ell}+\sum_{j=0}^{\ell-1}\left(t_j x_{1,j}\right)+x_3)},\ D_3 := g_2^{\tilde{r}},$$

where $\tilde{r} := r + \sum_{i=1}^{\ell} r^{(j)}$.

Suppose that $dk_{\mathtt{I},t_0} = (R_0^{(y)}, R_0^{(x)}, D_1, D_1', D_2, D_2', D_3)$ and $C = (C_0, C_1, C_2,\ C_3, \mathtt{tag})$ are correctly generated. Then, we have

$$\begin{aligned}
&\frac{C_0 e(C_3, D_3)}{e(C_1, D_1^{\mathtt{tag}}D_1')e(C_2, D_2^{\mathtt{tag}}D_2')} \\
&\quad = Me(g_1, g_2)^{(-x_0\alpha+y_0)s} \\
&\qquad \cdot \frac{e(g_1^{s(\sum_{j=0}^{\ell-1} t_j(-x_{1,j}\alpha+y_{1,j})+\mathtt{I}(-x_{1,\ell}\alpha+y_{1,\ell})+\mathtt{tag}(-x_2\alpha+y_2)-x_3\alpha+y_3)}, g_2^{\tilde{r}})}{e(g_1^s, g_2^{y_2\tilde{r}\mathtt{tag}+y_0+\tilde{r}(\mathtt{I}y_{1,\ell}+\sum_{j=0}^{\ell-1}\left(t_j y_{1,j}\right)+y_3)})e(g_1^{\alpha s}, g_2^{-x_2\tilde{r}\mathtt{tag}-x_0-\tilde{r}(\mathtt{I}x_{1,\ell}+\sum_{j=0}^{\ell-1}\left(t_j x_{1,j}\right)+x_3)})} \\
&\quad = Me(g_1, g_2)^{(-x_0\alpha+y_0)s}\frac{1}{e(g_1^s, g_2^{y_0})e(g_1^{\alpha s}, g_2^{-x_0})} = M.
\end{aligned}$$

We obtain the following theorem. The proof is postponed to Section 5.

Theorem 1. *If the SXDH assumption holds, then the resulting ℓ-level hierarchical IKE scheme Π_{IKE} is IND-KE-CPA secure.*

4.1 Parameters Evaluation and Comparison

First, we show the parameter sizes and computational costs of our hierarchical IKE scheme in Table 1.

Also, an efficiency comparison between our IKE scheme and the existing IKE schemes [21, 35] is given in Table 2. In fact, the WLC+08 scheme [35] has the threshold property and does not have a hierarchical structure, and therefore, we set the threshold value is one in the WLC+08 scheme and the hierarchy depth is one in the HHSI05 scheme [21] and our scheme for the fair comparison. The HHSI05 scheme meets the IND-KE-CCA security, however the scheme is secure only in the random oracle model (ROM). Both the WLC+08 scheme and ours meet the IND-KE-CPA security in the standard model (i.e. without random oracles). Although assumptions behind these schemes (i.e. the computational bilinear Diffie–Hellman (CBDH), decisional bilinear Diffie–Hellman (DBDH),[4] and SXDH assumptions) are different, they all are static and

[4]The formal definitions of the CBDH and DBDH assumptions are given in Appendix A.

Table 1: Parameters evaluation of our ℓ-level hierarchical IKE scheme.

$\#pp$	$\#dk$	$\#hk_\ell$	$\#hk_i$	$\#C$
$(\ell+5)\|\mathbb{G}_1\|+(2\ell+7)\|\mathbb{G}_2\|+\|\mathbb{G}_T\|$	$7\|\mathbb{G}_2\|$	$5\|\mathbb{G}_2\|$	$(2i+7)\|\mathbb{G}_2\|$	$3\|\mathbb{G}_1\|+\|\mathbb{G}_T\|+\|\mathbb{Z}_p\|$

Encryption Cost	Decryption Cost	Assumption
$[0,0,\ell+4,1]$	$[3,0,2,0]$	SXDH

$\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ are cyclic groups of order p, and $|\mathbb{G}_1|$, $|\mathbb{G}_2|$, and $|\mathbb{G}_T|$ denote the bit-length of a group element in $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$, respectively. $|\mathbb{Z}_p|$ also denotes the bit-length of an element in $\mathbb{Z}_p$. $\#pp$, $\#dk$, $\#hk_\ell$, $\#hk_i$, and $\#C$ denote sizes of public parameters, decryption keys, ℓ-th level helper keys, i-th level helper keys ($1 \le i \le \ell-1$), and ciphertexts, respectively. In computational cost analysis, $[\cdot,\cdot,\cdot,\cdot]$ means the number of [pairing, multi-exponentiation, regular exponentiation, fixed-based exponentiation]. For comparison we mention that relative costs for the various operations are as follows: [pairing$\approx$ 5, multi-exp$\approx$ 1.5, regular-exp:= 1, fixed-based-exp$\ll$ 0.2].

Table 2: Efficiency comparison between our construction and existing schemes.

Scheme	$\#pp$	$\#dk$	$\#hk$	$\#C$
HHSI05 [21] ($\ell=1$)	$2\|\mathbb{G}_p\|$	$3\|\mathbb{G}_p\|$	$\|\mathbb{G}_p\|$	$3\|\mathbb{G}_p\|+\|M\|+\|r\|$
WLC+08 [35]	$(2n+5)\|\mathbb{G}_p\|$	$4\|\mathbb{G}_p\|$	$2\|\mathbb{G}_p\|$	$3\|\mathbb{G}_p\|+\|\mathbb{G}_T\|$
Ours ($\ell=1$)	$6\|\mathbb{G}_1\|+9\|\mathbb{G}_2\|+\|\mathbb{G}_T\|$	$7\|\mathbb{G}_2\|$	$5\|\mathbb{G}_2\|$	$3\|\mathbb{G}_1\|+\|\mathbb{G}_T\|+\|\mathbb{Z}_p\|$

Scheme	Encryption Cost	Decryption Cost	Assumption
HHSI05 [21] ($\ell=1$)	$[1,0,2,1]$	$[4,0,2,1]$	CBDH (in ROM)
WLC+08 [35]	$[0,1,3,1]$	$[3,0,0,0]$	DBDH
Ours ($\ell=1$)	$[0,0,5,1]$	$[3,0,2,0]$	SXDH

The notation used here is almost the same as that in Table 1. $\#hk$ denotes the helper-key size, and $|\mathbb{G}_p|$ denotes the bit-length of a group element in a source group $\mathbb{G}_p$ in the symmetric setting. $|M|$ denotes the bit-length of plaintexts. r is a randomness that depends on the security parameter, and $|r|$ denotes its bit-length. n denotes the bit-length of identities in the scheme.

simple. We emphasize that the threshold structure does not strengthen the underlying DBDH assumption of the WLC+08 scheme since the structure was realized via only threshold secret sharing techniques [4, 29]. Note that we do not take into account the parallel IKE scheme [34] since the model of the scheme is slightly different from those of the above schemes. However, the public-parameter size of the parallel IKE scheme also depends on the size of its identity space, and we mention that this is due to the underlying Waters IBE [32], not due to the parallel property.

As can be seen, we first achieve the IKE scheme with constant-size parameters in the standard model. Again, we also get the first IKE scheme in the hierarchical setting without random oracles.

5 Proof of Security

We describe how semi-functional ciphertexts and secret keys are generated as follows.

Semi-functional Ciphertext: Parse a normal ciphertext C as $(C_0, C_1, C_2, C_3, \mathtt{tag})$. A semi-functional ciphertext $\widetilde{C} := (\tilde{C}_0, \tilde{C}_1, \tilde{C}_2, \tilde{C}_3, \widetilde{\mathtt{tag}})$ is computed as follows:

$$\tilde{C}_0 := C_0 e(g_1, g_2)^{-x_0\mu} = M e(g_1, g_2)^{-x_0(\alpha s+\mu)+y_0 s},$$

$$
\begin{aligned}
\tilde{C}_1 &:= C_1,\\
\tilde{C}_2 &:= C_2 g_1^{\mu} = g_1^{\alpha s+\mu},\\
\tilde{C}_3 &:= C_3\Big((g_1^{x_{1,\ell}})^{\mathtt{I}}\prod_{j=0}^{\ell-1}\big((g_1^{x_{1,j}})^{t_j}\big)(g_1^{x_2})^{\mathtt{tag}}g_1^{x_3}\Big)^{-\mu}\\
&= C_3 g_1^{-\mu(\mathtt{I}x_{1,\ell}+\sum_{j=0}^{\ell-1}(t_j x_{1,j})+x_2\mathtt{tag}+x_3)}\\
&= g_1^{-(\alpha s+\mu)(\mathtt{I}x_{1,\ell}+\sum_{j=0}^{\ell-1}(t_j x_{1,j})+x_2\mathtt{tag}+x_3)} g_1^{s(\mathtt{I}y_{1,\ell}+\sum_{j=0}^{\ell-1}(t_j y_{1,j})+y_2\mathtt{tag}+y_3)},
\end{aligned}
$$

and $\widetilde{\mathtt{tag}} := \mathtt{tag}$, where $\mu \xleftarrow{\$} \mathbb{Z}_p^*$.

Semi-functional Decryption and Helper Key: Parse a normal helper key $hk_{\mathtt{I},t_i}^{(i)}$ as $(R_i, D_1, D_1', D_2, D_2', D_3, \{(K_j, K_j')\}_{j=0}^{i-1})$. A semi-functional helper key $\widetilde{hk}_{\mathtt{I},t_i}^{(i)} := (\tilde{R}_i, \tilde{D}_1, \tilde{D}_1', \tilde{D}_2, \tilde{D}_2', \tilde{D}_3, \{(\tilde{K}_j, \tilde{K}_j')\}_{j=0}^{i-1})$ is computed as follows: $\tilde{R}_i^{(y)} := R_i^{(y)}$, $\tilde{R}_i^{(x)} := R_i^{(x)}$,

$$
\begin{aligned}
\tilde{D}_1 &:= D_1 g_2^{\gamma} = g_2^{y_2 r+\gamma},\\
\tilde{D}_1' &:= D_1' g_2^{\gamma\phi} = g_2^{y_0+r(\mathtt{I}y_{1,\ell}+\sum_{j=i}^{\ell-1}(t_j y_{1,j})+y_3)+\gamma\phi},\\
\tilde{D}_2 &:= D_2 g_2^{-\frac{\gamma}{\alpha}} = g_2^{-rx_2-\frac{\gamma}{\alpha}},\\
\tilde{D}_2' &:= D_2' g_2^{-\frac{\gamma\phi}{\alpha}} = g_2^{-x_0-r(\mathtt{I}x_{1,\ell}+\sum_{j=i}^{\ell-1}(t_j x_{1,j})+x_3)-\frac{\gamma\phi}{\alpha}},\\
\tilde{D}_3 &:= D_3,\\
\tilde{K}_j &:= K_j g_2^{\gamma\phi_j} = g_2^{ry_{1,j}+\gamma\phi_j}\ (0 \le j \le i-1),\\
\tilde{K}_j' &:= K_j' g_2^{-\frac{\gamma\phi_j}{\alpha}} = g_2^{-rx_{1,j}-\frac{\gamma\phi_j}{\alpha}}\ (0 \le j \le i-1),
\end{aligned}
$$

where $\phi, \{\phi_j\}_{j=0}^{i-1} \xleftarrow{\$} \mathbb{Z}_p$ and $\gamma \xleftarrow{\$} \mathbb{Z}_p^*$. Note that $hk_{\mathtt{I},t_0}^{(0)}$ means $dk_{\mathtt{I},t_0}$ for any $t_0 \in \mathcal{T}_0$. In particular, $\widetilde{hk}_{\mathtt{I},t_0}^{(0)}$ $(= \widetilde{dk}_{\mathtt{I},t_0})$ is called a semi-functional decryption key. We also note that in order to generate the semi-functional decryption or helper key, $g_2^{\frac{1}{\alpha}}$ is needed in addition to the public parameter.

A semi-functional ciphertext can be decrypted with a normal key. This fact can be easily checked by

$$
\frac{e(g_1,g_2)^{-x_0\mu}e(g_1^{-\mu(\mathtt{I}x_{1,\ell}+\sum_{j=0}^{\ell-1}(t_j x_{1,j})+x_2\mathtt{tag}+x_3)}, D_3)}{e(g_1^{\mu}, D_2^{\mathtt{tag}}D_2')} = 1.
$$

Also, a normal ciphertext can be decrypted with a semi-functional decryption key since it holds

$$
e(C_1, g_2^{\gamma\mathtt{tag}}g_2^{\gamma\phi})e(C_2, g_2^{-\frac{\gamma}{\alpha}\mathtt{tag}}g_2^{-\frac{\gamma\phi}{\alpha}}) = 1.
$$

A helper or decryption key obtained by running the Δ-Gen and Upd algorithms with a semi-functional helper key is also semi-functional.

Proof of Theorem 1. Based on [24, 27], we prove the theorem through a sequence of games. We first define the following games:

$\mathsf{Game}_{\mathsf{Real}}$: This is the same as the IND-KE-CPA game described in Section 3.

Game_0: This is the same as $\mathsf{Game}_{\mathsf{Real}}$ except that the challenge ciphertext is semi-functional.

Game_k $(1 \le k \le q)$**:** This is the same as Game_0 except for the following modification: Let q be the maximum number of identities issued to the KG or KI oracles, and $\mathtt{I}_i$ $(1 \le i \le q)$ be an i-th identity issued to the oracles. If queries regarding the first k identities $\mathtt{I}_1, \ldots, \mathtt{I}_k$ are issued, then semi-functional decryption and/or helper keys are returned. The rest of keys (i.e., keys regarding $\mathtt{I}_{k+1}, \ldots, \mathtt{I}_q$) are normal.

$\mathsf{Game}_{\mathsf{Final}}$: This is the same as Game_q except that the challenge ciphertext is a semi-functional one of a random element of $\mathbb{G}_T$.

Let S_{Real}, S_k $(0 \le k \le q)$, and S_{Final} be the probabilities that the event $b' = b$ occurs in $\mathsf{Game}_{\mathsf{Real}}$, Game_k, and $\mathsf{Game}_{\mathsf{Final}}$, respectively. Then, we have

$$Adv^{IND\text{-}KE\text{-}CPA}_{\Pi_{IKE},\mathcal{A}}(\lambda,\ell) \le |S_{\mathsf{Real}} - S_0| + \sum_{i=1}^{q} |S_{i-1} - S_i| + |S_q - S_{\mathsf{Final}}| + |S_{\mathsf{Final}} - \frac{1}{2}|.$$

The rest of the proof follows from the following lemmas.

Lemma 1. *If the DDH1 assumption holds, then it holds that* $|S_{\mathsf{Real}} - S_0| \le 2Adv^{DDH1}_{\mathcal{G},\mathcal{B}}(\lambda)$.

Proof. At the beginning, a PPT adversary $\mathcal{B}$ receives an instance $(g_1, g_1^{c_1}, g_1^{c_2}, g_2, T)$ of the DDH1 problem. Then, $\mathcal{B}$ randomly chooses $x_0, y_0, \{(x_{1,j}, y_{1,j})\}_{j=0}^{\ell}, x_2, y_2, x_3, y_3 \xleftarrow{\$} \mathbb{Z}_p$, and creates

$$z := e(g_1^{c_1}, g_2)^{-x_0} e(g_1, g_2)^{y_0},\ u_{1,j} := (g_1^{c_1})^{-x_{1,j}} g_1^{y_{1,j}}\ (0 \le j \le \ell),\ w_1 := (g_1^{c_1})^{-x_2} g_1^{y_2},\ h_1 := (g_1^{c_1})^{-x_3} g_1^{y_3}.$$

$\mathcal{B}$ sends $pp := (g_1, g_1^{\alpha}, \{u_{1,j}\}_{j=0}^{\ell}, w_1, h_1, g_2, \{(g_2^{x_{1,j}}, g_2^{y_{1,j}})\}_{j=0}^{\ell}, g_2^{x_2}, g_2^{x_3}, g_2^{y_2}, g_2^{y_3}, z)$ to $\mathcal{A}$. Note that $\mathcal{B}$ knows a master key $mk := (x_0, y_0)$ and we implicitly set $\alpha := c_1$.

$\mathcal{B}$ can simulate the KG and KI oracles since $\mathcal{B}$ knows the master key.

In the challenge phase, $\mathcal{B}$ receives $(M_0^*, M_1^*, \mathtt{I}^*, \mathtt{time}^*)$ from $\mathcal{A}$. $\mathcal{B}$ chooses $d \xleftarrow{\$} \{0,1\}$. $\mathcal{B}$ chooses $\mathtt{tag}^* \xleftarrow{\$} \mathbb{Z}_p$, and let $t_j^* := T_j(\mathtt{time}^*)$ $(0 \le j \le \ell - 1)$. $\mathcal{B}$ computes

$$\begin{aligned}
&C_0^* := M_d^* e(T, g_2)^{-x_0} e(g_1^{c_2}, g_2)^{y_0},\ C_1^* := g_1^{c_2},\ C_2^* := T,\\
&C_3^* := T^{-\mathtt{I}^* x_{1,\ell} - \sum_{j=0}^{\ell-1}(t_j^* x_{1,j}) - x_2 \mathtt{tag}^* - x_3} (g_1^{c_2})^{\mathtt{I}^* y_{1,\ell} + \sum_{j=0}^{\ell-1}(t_j^* y_{1,j}) + y_2 \mathtt{tag}^* + y_3}.
\end{aligned}$$

$\mathcal{B}$ sends $C^* := (C_0^*, C_1^*, C_2^*, C_3^*, \mathtt{tag}^*)$ to $\mathcal{A}$.

If $b = 0$, then the above ciphertext is normal by setting $s := c_2$. If $b = 1$, then the above ciphertext is semi-functional since it holds

$$\begin{aligned}
C_0^* &= M_d^* e(g_1, g_2)^{-x_0(c_1 c_2 + \mu) + y_0 c_2} = M_d^* e(g_1, g_2)^{-x_0(\alpha s + \mu) + y_0 s},\\
C_2^* &= g_1^{c_1 c_2 + \mu} = g_1^{\alpha s + \mu},\\
C_3^* &= g_1^{-(c_1 c_2 + \mu)(\mathtt{I}^* x_{1,\ell} + \sum_{j=0}^{\ell-1}(t_j^* x_{1,j}) + x_2 \mathtt{tag}^* + x_3)} g_1^{c_2(\mathtt{I}^* y_{1,\ell} + \sum_{j=0}^{\ell-1}(t_j^* y_{1,j}) + y_2 \mathtt{tag}^* + y_3)}\\
&= g_1^{-(\alpha s + \mu)(\mathtt{I}^* x_{1,\ell} + \sum_{j=0}^{\ell-1}(t_j^* x_{1,j}) + x_2 \mathtt{tag}^* + x_3)} g_1^{s(\mathtt{I}^* y_{1,\ell} + \sum_{j=0}^{\ell-1}(t_j^* y_{1,j}) + y_2 \mathtt{tag}^* + y_3)}.
\end{aligned}$$

After receiving d' from $\mathcal{A}$, $\mathcal{B}$ sends $b' = 1$ to the challenger of the DDH1 problem if $d' = d$. Otherwise, $\mathcal{B}$ sends $b' = 0$ to the challenger. □

Lemma 2. *For every* $k \in \{1, 2, \ldots, q\}$, *if the DDH2 assumption holds, then it holds that* $|S_{k-1} - S_k| \le 2Adv^{DDH2}_{\mathcal{G},\mathcal{B}}(\lambda)$.

Proof. At the beginning, a PPT adversary $\mathcal{B}$ receives an instance $(g_1, g_2, g_2^{c_1}, g_2^{c_2}, T)$ of the DDH2 problem. Then, $\mathcal{B}$ randomly chooses $x'_0, y_0, \{(x'_{1,j}, y'_{1,j}, y''_{1,j})\}_{j=0}^{\ell}, x'_2, x'_3, y'_3, y''_3 \xleftarrow{\$} \mathbb{Z}_p$ and $\alpha \xleftarrow{\$} \mathbb{Z}_p^{\times}$, and (implicitly) sets

$$\begin{aligned}
&x_0 := \frac{x'_0 + y_0}{\alpha},\quad x_{1,j} := \frac{x'_{1,j} + y_{1,j}}{\alpha},\ \text{where } y_{1,j} := y'_{1,j} + c_2 y''_{1,j}\ (0 \le j \le \ell),\\
&x_2 := \frac{x'_2 + c_2}{\alpha},\quad y_2 := c_2,\quad x_3 := \frac{x'_3 + y_3}{\alpha},\ \text{where } y_3 := y'_3 + c_2 y''_3.
\end{aligned}$$

$\mathcal{B}$ creates

$$z := e(g_1, g_2)^{-x'_0},\quad u_{1,j} := g_1^{-x'_{1,j}}\ (0 \le j \le \ell),\quad w_1 := g_1^{-x'_2},\quad h_1 := g_1^{-x'_3},$$

$$g_2^{x_{1,j}} := g_2^{\frac{x'_{1,j}+y'_{1,j}}{\alpha}}(g_2^{c_2})^{\frac{y''_{1,j}}{\alpha}}\ (0 \le j \le \ell),\quad g_2^{y_{1,j}} := g_2^{y'_{1,j}}(g_2^{c_2})^{y''_{1,j}}\ (0 \le j \le \ell),$$
$$g_2^{x_2} := g_2^{\frac{x'_2}{\alpha}}(g_2^{c_2})^{\frac{1}{\alpha}},\quad g_2^{y_2} := g_2^{c_2},\quad g_2^{x_3} := g_2^{\frac{x'_3+y'_3}{\alpha}}(g_2^{c_2})^{\frac{y''_3}{\alpha}},\quad g_2^{y_3} := g_2^{y'_3}(g_2^{c_2})^{y''_3}.$$

$\mathcal{B}$ sends $pp := (g_1, g_1^\alpha, \{u_{1,j}\}_{j=0}^{\ell}, w_1, h_1, g_2, \{(g_2^{x_{1,j}}, g_2^{y_{1,j}})\}_{j=0}^{\ell}, g_2^{x_2}, g_2^{y_2}, g_2^{x_3}, g_2^{y_3}, z)$ to $\mathcal{A}$. Note that $\mathcal{B}$ knows a master key $mk := (x_0, y_0)$.

We show how $\mathcal{B}$ simulates the KG and KI oracles. Let $\mathtt{I}_i$ $(1 \le i \le q)$ be an i-th identity issued to the oracles. Without loss of generality, we consider $\mathcal{A}$ issues all identities $\mathtt{I}_i \neq \mathtt{I}^*$ to the KG oracle, and issues only queries regarding $\mathtt{I}^*$ to the KI oracle.

***KG* oracle.** $\mathcal{B}$ creates $k-1$ semi-functional decryption and helper keys, and embeds T into the k-th keys. The rest of keys are normal.

Case $i < k$: After receiving $\mathtt{I}_i$, $\mathcal{B}$ creates and returns semi-functional keys. Since $\mathcal{B}$ knows the master key and α, $\mathcal{B}$ can create both normal and semi-functional keys.

Case $i = k$: After receiving $\mathtt{I}_k$, $\mathcal{B}$ creates semi functional keys by embedding T as follows: $\mathcal{B}$ chooses $\beta_0^{(y)}, \ldots, \beta_{\ell-1}^{(y)}, \beta_0^{(x)}, \ldots, \beta_{\ell-1}^{(x)} \xleftarrow{\$} \mathbb{Z}_p$, and sets $B^{(y)} := \sum_{j=0}^{\ell-1}\beta_j^{(y)}$ and $B^{(x)} := \sum_{j=0}^{\ell-1}\beta_j^{(x)}$. $\mathcal{B}$ computes

$$\begin{aligned}
R_j^{(y)} &:= g_2^{-\beta_j^{(y)}}\ (0 \le j \le \ell-1),\\
R_j^{(x)} &:= g_2^{-\beta_j^{(x)}}\ (0 \le j \le \ell-1),\\
D_1 &:= T,\\
D'_1 &:= g_2^{y_0+B^{(y)}}(g_2^{c_1})^{\mathtt{I}_k y'_{1,\ell}+y'_3}T^{\mathtt{I}_k y''_{1,\ell}+y''_3},\\
D_2 &:= \left((g_2^{c_1})^{x'_2}T\right)^{-\frac{1}{\alpha}},\\
D'_2 &:= g_2^{-\frac{x'_0}{\alpha}-B^{(x)}}(g_2^{c_1})^{-\frac{\mathtt{I}_k(x'_{1,\ell}+y'_{1,\ell})+x'_3+y'_3}{\alpha}}g_2^{-\frac{y_0}{\alpha}}T^{-\frac{\mathtt{I}_k y''_{1,\ell}+y''_3}{\alpha}},\\
D_3 &:= g_2^{c_1},\\
K_j &:= (g_2^{c_1})^{y'_{1,j}}(T)^{y''_{1,j}}\ (0 \le j \le \ell-1),\\
K'_j &:= (g_2^{c_1})^{-\frac{x'_{1,j}+y'_{1,j}}{\alpha}}T^{-\frac{y''_{1,j}}{\alpha}}\ (0 \le j \le \ell-1).
\end{aligned}$$

$\mathcal{B}$ sets $dk_{\mathtt{I},0} := (R_0^{(y)}, R_0^{(x)})$, $hk_{\mathtt{I},0}^{(i)} := (R_i^{(y)}, R_i^{(x)})$ $(1 \le i \le \ell-1)$, $hk_{\mathtt{I},0}^{(\ell)} := (D_1, D'_1, D_2, D'_2, D_3, \{(K_j, K'_j)\}_{j=0}^{\ell-1})$. If $b = 0$, then it is easy to see that the above keys are normal by setting $r := c_1$. If $b = 1$, then the above ciphertext is semi-functional since it holds

$$\begin{aligned}
D_1 :=& T = g_2^{c_1c_2+\gamma} = g_2^{y_2 r+\gamma},\\
D'_1 :=& g_2^{y_0+B^{(y)}}(g_2^{c_1})^{\mathtt{I}_k y'_{1,\ell}+y'_3}T^{\mathtt{I}_k y''_{1,\ell}+y''_3}\\
=& g_2^{y_0+B^{(y)}+c_1(\mathtt{I}_k(y'_{1,\ell}+c_2y''_{1,\ell})+y'_3+c_2y''_3)}g_2^{\gamma(\mathtt{I}_k y''_{1,\ell}+y''_3)} = g_2^{y_0+B^{(y)}+r(\mathtt{I}_k y_{1,\ell}+y_3)}g_2^{\gamma\phi},\\
D_2 :=& \left((g_2^{c_1})^{x'_2}T\right)^{-\frac{1}{\alpha}} = g_2^{-\frac{c_1(x'_2+c_2)}{\alpha}}g_2^{-\frac{\gamma}{\alpha}} = g_2^{-rx_2}g_2^{-\frac{\gamma}{\alpha}},\\
D'_2 :=& g_2^{-\frac{x'_0}{\alpha}-B^{(x)}}(g_2^{c_1})^{-\frac{\mathtt{I}_k(x'_{1,\ell}+y'_{1,\ell})+x'_3+y'_3}{\alpha}}g_2^{-\frac{y_0}{\alpha}}T^{-\frac{\mathtt{I}_k y''_{1,\ell}+y''_3}{\alpha}}\\
=& g_2^{-B^{(x)}-\frac{(x'_0+y_0)+c_1(\mathtt{I}_k(x'_{1,\ell}+y'_{1,\ell}+c_2y''_{1,\ell})+(x'_3+y'_3+c_2y''_3))}{\alpha}}g_2^{-\frac{\gamma(\mathtt{I}_k y''_{1,\ell}+y''_3)}{\alpha}}\\
=& g_2^{-x_0-B^{(x)}-r(\mathtt{I}_k x_{1,\ell}+x_3)}g_2^{-\frac{\gamma\phi}{\alpha}},\\
K_j :=& (g_2^{c_1})^{y'_{1,j}}(T)^{y''_{1,j}} = g_2^{c_1(y'_{1,j}+c_2y''_{1,j})}g_2^{\gamma y''_{1,j}} = g_2^{ry_{1,j}}g_2^{\gamma\phi_j}\ (0 \le j \le \ell-1),\\
K'_j :=& (g_2^{c_1})^{-\frac{x'_{1,j}+y'_{1,j}}{\alpha}}T^{-\frac{y''_{1,j}}{\alpha}}
\end{aligned}$$

$$=g_2^{-\frac{c_1(x'_{1,j}+y'_{1,j}+c_2y''_{1,j})}{\alpha}} g_2^{-\frac{\gamma y''_{1,j}}{\alpha}} = g_2^{-rx_{1,j}} g_2^{-\frac{\gamma\phi_j}{\alpha}} \ (0 \le j \le \ell-1),$$

where $T := g_2^{c_1c_2+\gamma}$, $r := c_1$, $\phi := \mathtt{I}_k y''_{1,\ell} + y''_3$, and $\phi_j := y''_{1,j}$ $(0 \le j \le \ell-1)$. Since $y''_{1,j}$ and y''_3 are chosen uniformly at random, ϕ and ϕ_j are also uniformly distributed.

Case $i > k$: After receiving $\mathtt{I}_i$, $\mathcal{B}$ creates and returns normal keys by using the master key.

***KI* oracle** Without loss of generality, suppose that $\mathcal{A}$ issues $k-1$ identities $\mathtt{I}_1, \ldots, \mathtt{I}_{k-1}$ to the *KG* oracle, and then issues a query $(i, \mathtt{I}^*, \mathtt{time})$ (i.e., $\mathtt{I}^* = \mathtt{I}_k$) to the *KI* oracle. Note that for some special level $j \in \{0, \ldots, \ell\}$, $\mathcal{A}$ cannot issue $\mathtt{time}$ such that $T_i(\mathtt{time}) = T_i(\mathtt{time}^*)$ if $i < j$ ($\mathcal{B}$ does not need to know which level would be the special one in advance). $\mathcal{B}$ creates and stores decryption and helper keys $(dk_{\mathtt{I}^*,0}, hk^{(1)}_{\mathtt{I}^*,0}, \ldots, hk^{(\ell)}_{\mathtt{I}^*,0})$ as in the case $i = k$ of the *KG* oracle. Then, $\mathcal{B}$ repeatedly runs $\delta^{(k-1)}_{t_{k-1}} \leftarrow \Delta\text{-}\mathsf{Gen}(hk^{(k)}_{\mathtt{I}^*,t_k}, \mathtt{time}^*)$ and $hk^{(k-1)}_{\mathtt{I}^*,t^*_{k-1}} \mathsf{Upd}(hk^{(k-1)}_{\mathtt{I}^*,0}, \delta^{(k-1)}_{t_{k-1}})$ for $k = \ell, \ldots, i+1$, where $t_\ell := 0$ and $t_k := T_k(\mathtt{time})$ $(i \le k \le \ell-1)$. $\mathcal{B}$ returns $hk^{(i)}_{\mathtt{I}^*,t_i}$ to $\mathcal{A}$. Note that from the second query for $\mathtt{I}^*$, $\mathcal{B}$ answers queries by using the stored keys.

It is obvious that the returned key is an well-formed normal key if $b = 0$. We show that the returned key is semi-functional if $b = 1$. Since $(dk_{\mathtt{I}^*,0}, hk^{(1)}_{\mathtt{I}^*,0}, \ldots, hk^{(\ell)}_{\mathtt{I}^*,0})$ is generated as in the case $i = k$ of the *KG* oracle, the forms of $hk_{\mathtt{I}^*,t_i} = (R^{(y)}_i, R^{(x)}_i, D_1, D'_1, D_2, D'_2, D_3, \{K_k, K'_k\}_{k=0}^{i-1})$ are as follows.

$$\begin{aligned}
R^{(y)}_i =& g_2^{-\beta^{(y)}_i}, \quad R^{(x)}_i = g_2^{\beta^{(x)}_i},\\
D_1 =& g_2^{(c_1+\hat{r})c_2+\gamma} = g_2^{y_2 r+\gamma},\\
D'_1 =& g_2^{y_0+\sum_{k=0}^{i-1}\beta^{(y)}_k+(c_1+\hat{r})(\mathtt{I}^*(y'_{1,\ell}+c_2y''_{1,\ell})+\sum_{k=i-1}^{\ell-1}t_k(y'_{1,k}+c_2y''_{1,k})+y'_3+c_2y''_3)}\\
&\qquad \cdot g_2^{\gamma(\mathtt{I}^*y''_{1,\ell}+\sum_{k=i-1}^{\ell-1}t_ky''_{1,k}+y''_3)}\\
=& g_2^{y_0+\sum_{k=0}^{i-1}\beta^{(y)}_k+r(\mathtt{I}^*y_{1,\ell}+\sum_{k=i-1}^{\ell-1}t_ky_{1,k}+y_3)} g_2^{\gamma\phi},\\
D_2 =& g_2^{-\frac{(c_1+\hat{r})(x'_2+c_2)}{\alpha}} g_2^{-\frac{\gamma}{\alpha}} = g_2^{-rx_2} g_2^{-\frac{\gamma}{\alpha}},\\
D'_2 =& g_2^{-\frac{x'_0+y_0}{\alpha}-\sum_{k=0}^{i-1}\beta^{(x)}_k-\frac{(c_1+\hat{r})(\mathtt{I}^*(x'_{1,\ell}+y'_{1,\ell}+c_2y''_{1,\ell})+\sum_{k=i-1}^{\ell-1}t_k(x'_{1,k}+y'_{1,k}+c_2y''_{1,k})+(x'_3+y'_3+c_2y''_3))}{\alpha}} g_2^{-\frac{\gamma(\mathtt{I}^*y''_{1,\ell}+\sum_{k=i-1}^{\ell-1}t_ky''_{1,k}+y''_3)}{\alpha}}\\
=& g_2^{-x_0-\sum_{k=0}^{i-1}\beta^{(x)}_k-r(\mathtt{I}^*x_{1,\ell}+\sum_{k=i-1}^{\ell-1}t_kx_{1,k}+x_3)} g_2^{-\frac{\gamma\phi}{\alpha}},\\
K_k =& g_2^{(c_1+\hat{r})(y'_{1,k}+c_2y''_{1,k})} g_2^{\gamma y''_{1,k}} = g_2^{ry_{1,k}} g_2^{\gamma\phi_k} \ (0 \le k \le i-1),\\
K'_k =& g_2^{-\frac{(c_1+\hat{r})(x'_{1,k}+y'_{1,k}+c_2y''_{1,k})}{\alpha}} g_2^{-\frac{\gamma y''_{1,k}}{\alpha}} = g_2^{-rx_{1,k}} g_2^{-\frac{\gamma\phi_k}{\alpha}} \ (0 \le k \le i-1),
\end{aligned}$$

where γ comes from $T = g_2^{c_1c_2+\gamma}$, $\hat{r}$ is randomness due to the re-randomization procedure in the $\Delta\text{-}\mathsf{Gen}$ algorithm, $r := c_1 + \hat{r}$, $\phi := \mathtt{I}^*y''_{1,\ell} + \sum_{k=i-1}^{\ell-1} t_k y''_{1,k} + y''_3$, and $\phi_j := y''_{1,k}$ $(i-1 \le k \le \ell-1)$.

We have to pay attention to a query $(i, \mathtt{I}^*, \mathtt{time})$ such that $i > j$ and $T_i(\mathtt{time}) = T_j(\mathtt{time}^*)$. In the case above, $\mathcal{A}$ can derive

$$\overline{dk}_{\mathtt{I}^*,T_0(\mathtt{time}^*)} = (R^{(y)}_0,\ R^{(x)}_0,\ D_1,\ D'_1 \cdot g_2^{\beta^{(y)}_j},\ D_2,\ D'_2 \cdot g_2^{-\beta^{(x)}_j},\ D_3),$$

from $hk_{\mathtt{I}^*,T_i(\mathtt{time})}$. Namely, $\mathcal{A}$ can obtain a decryption key for $\mathtt{I}^*$ and $\mathtt{time}^*$ with noises $\beta^{(y)}_j$ and $\beta^{(x)}_j$. Then, we have $\phi = \mathtt{I}^*y''_{1,\ell} + \sum_{k=0}^{\ell-1}(t^*_k y''_{1,k}) + y''_3$. This ϕ is the same as $\widetilde{\mathtt{tag}}^*$, which is defined in the challenge phase. Therefore, ϕ is not uniformly distributed from the viewpoint of $\mathcal{A}$, and the proof seem to fail. However, the simulation actually works well since we can observe the above simulation from another perspective: We regard $(\beta^{(y)}_j, \beta^{(x)}_j)$ as $(\beta'^{(y)}_j + \chi, \beta'^{(x)}_j + \frac{\chi}{\alpha})$. The above ϕ then turns to $\mathtt{I}^*y''_{1,\ell} + \sum_{k=0}^{\ell-1}(t^*_k y''_{1,k}) + y''_3 + \chi$ (and the noises turns to $\beta'^{(y)}_j$ and $\beta'^{(x)}_j$), and therefore such a collision of randomness never occurs since $\mathcal{A}$ never knows the values of $(\beta^{(y)}_j, \beta^{(x)}_j)$.

In the challenge phase, $\mathcal{B}$ receives $(M_0^*, M_1^*, \mathtt{I}^*, \mathtt{time}^*)$ from $\mathcal{A}$. $\mathcal{B}$ chooses $d \xleftarrow{\$} \{0,1\}$, and sets $t_j^* := T_j(\mathtt{time}^*)$ $(0 \le j \le \ell-1)$. However, $\mathcal{B}$ cannot create the semi-functional ciphertext for $\mathtt{I}^*$ without knowledge of c_2 (and hence $y_{1,j}$ $(0 \le j \le \ell)$ and y_3). To generate the semi-functional ciphertext without the knowledge, $\mathcal{B}$ sets

$$\widetilde{\mathtt{tag}}^* := -\sum_{j=0}^{\ell-1}(t_j^* y''_{1,j}) - \mathtt{I}^* y''_{1,\ell} - y''_3.$$

Since $y''_{1,0}, \ldots, y''_{1,\ell}$ and y''_3 are chosen uniformly at random, probability distribution of $\widetilde{\mathtt{tag}}^*$ is also uniformly at random from $\mathcal{A}$'s view.[5] Then, $\mathcal{B}$ chooses $s \xleftarrow{\$} \mathbb{Z}_p$ and $\mu \xleftarrow{\$} \mathbb{Z}_p^*$, and computes

$$\begin{aligned}
\tilde{C}_0^* :=& M_d^* z^s e(g_1,g_2)^{-x_0\mu} = M_d^* e(g_1,g_2)^{-x_0(\alpha s+\mu)+y_0 s},\\
\tilde{C}_1^* :=& g_1^s,\\
\tilde{C}_2^* :=& g_1^{\alpha s+\mu}\\
\tilde{C}_3^* :=& \Big(\prod_{j=0}^{\ell-1}(u_{1,j}^{t_j^*}) u_{1,\ell}^{\mathtt{I}^*} w_1^{\widetilde{\mathtt{tag}}^*} h_1\Big)^s g_1^{-\frac{\mu}{\alpha}(\sum_{j=0}^{\ell-1}(t_j^*(x'_{1,j}+y'_{1,j}))+\mathtt{I}^*(x'_{1,\ell}+y'_{1,\ell})+x'_2\widetilde{\mathtt{tag}}^*+x'_3+y'_3)}\\
=& \Big(\prod_{j=0}^{\ell-1}(u_{1,j}^{t_j^*}) u_{1,\ell}^{\mathtt{I}^*} w_1^{\widetilde{\mathtt{tag}}^*} h_1\Big)^s\\
&\qquad \cdot g_1^{-\frac{\mu}{\alpha}(\sum_{j=0}^{\ell-1}(t_j^*(x'_{1,j}+y'_{1,j}))+\mathtt{I}^*(x'_{1,\ell}+y'_{1,\ell})+x'_2\widetilde{\mathtt{tag}}^*+x'_3+y'_3)} g_1^{-\frac{c_2\mu}{\alpha}(\sum_{j=0}^{\ell-1}(t_j^* y''_{1,j})+\mathtt{I}^* y''_{1,\ell}+\widetilde{\mathtt{tag}}^*+y''_3)}\\
=& \Big(\prod_{j=0}^{\ell-1}(u_{1,j}^{t_j^*}) u_{1,\ell}^{\mathtt{I}^*} w_1^{\widetilde{\mathtt{tag}}^*} h_1\Big)^s g_1^{\mu(\sum_{j=0}^{\ell-1}(t_j^* x_{1,j})+\mathtt{I}^* x_{1,\ell}+x_2\widetilde{\mathtt{tag}}^*+x_3)}.
\end{aligned}$$

$\mathcal{B}$ sends $\widetilde{C}^* := (\tilde{C}_0^*, \tilde{C}_1^*, \tilde{C}_2^*, \tilde{C}_3^*, \widetilde{\mathtt{tag}}^*)$ to $\mathcal{A}$.

After receiving d' from $\mathcal{A}$, $\mathcal{B}$ sends $b'=1$ to the challenger of the DDH2 problem if $d'=d$. Otherwise, $\mathcal{B}$ sends $b'=0$ to the challenger. □

Lemma 3. $|S_q - S_{\mathsf{Final}}| \le 2Adv_{\mathcal{G},\mathcal{B}}^{DDH1}(\lambda)$.

Proof. At the beginning, a PPT adversary $\mathcal{B}$ receives an instance $(g_1, g_1^{c_1}, g_1^{c_2}, g_2, T)$ of the DDH1 problem. Then, $\mathcal{B}$ randomly chooses $\{(x_{1,j}, y'_{1,j})\}_{j=0}^{\ell}, x_2, y'_2, x_3, y'_3 \xleftarrow{\$} \mathbb{Z}_p$ and $\alpha \xleftarrow{\$} \mathbb{Z}_p^*$, and (implicitly) sets

$$x_0 := c_1,\ y_0 := x_0\alpha + y'_0,\ y_{1,j} := x_{1,j}\alpha + y'_{1,j}\ (0 \le j \le \ell),\ y_2 := x_2\alpha + y'_2,\ y_3 := x_3\alpha + y'_3.$$

Then, $\mathcal{B}$ creates

$$z := e(g_1,g_2)^{y'_0},\ u_{1,j} := g_1^{y'_{1,j}}\ (0 \le j \le \ell),\ w_1 := g_1^{y'_2},\ h_1 := g_1^{y'_3}.$$

$\mathcal{B}$ sends $pp := (g_1, g_1^\alpha, \{u_{1,j}\}_{j=0}^{\ell}, w_1, h_1, g_2, \{(g_2^{x_{1,j}}, g_2^{y_{1,j}})\}_{j=0}^{\ell}, g_2^{x_2}, g_2^{x_3}, g_2^{y_2}, g_2^{y_3}, z)$ to $\mathcal{A}$. Note that $\mathcal{B}$ does not know a master key $mk := (x_0, y_0)$.

***KG* oracle.** When receiving $\mathtt{I}$ from $\mathcal{A}$, $\mathcal{B}$ first generates (initial) semi-functional keys as follows. $\mathcal{B}$ chooses $\beta_0^{(y)}, \ldots, \beta_{\ell-1}^{(y)}, \beta_0^{(x)}, \ldots, \beta_{\ell-1}^{(x)}, r, \phi', \phi'_0, \ldots, \phi'_{\ell-1} \xleftarrow{\$} \mathbb{Z}_p$ and $\gamma \xleftarrow{\$} \mathbb{Z}_p^*$, and (implicitly) set $B^{(y)} := \sum_{j=0}^{\ell-1}\beta_j^{(y)}$, $B^{(x)} := \sum_{j=0}^{\ell-1}\beta_j^{(x)}$, $\phi' := x_0 + r(\mathtt{I}x_{1,\ell} + x_3) + \frac{\gamma\phi}{\alpha}$, and $\phi'_j := rx_{1,j} + \frac{\gamma\phi_j}{\alpha}$ $(0 \le j \le \ell-1)$. We compute

$$\tilde{R}_j^{(y)} := g_2^{-\beta_j^{(y)}}\ (0 \le j \le \ell-1),$$

[5] The fact that the formula in such a form is uniformly distributed was traditionally studied in the context of unconditionally secure authentication protocols (e.g., [11, 23, 30]).

$$
\begin{aligned}
\tilde{R}_j^{(x)} :=& g_2^{\beta_j^{(x)}} \ (0 \le j \le \ell-1),\\
\tilde{D}_1 :=& g_2^{y_2 r+\gamma},\\
\tilde{D}'_1 :=& g_2^{y'_0+r(\mathtt{I}y'_{1,\ell}+y'_3)+\alpha\phi'} = g_2^{x_0\alpha+y'_0+r\left((x_{1,\ell}\alpha+y_{1,\ell})\mathtt{I}+x_3+y'_3\right)+\gamma\phi} = g_2^{y_0+r(y_{1,\ell}\mathtt{I}+y_3)+\gamma\phi},\\
\tilde{D}_2 :=& g_2^{-rx_2-\frac{\gamma}{\alpha}}\\
\tilde{D}'_2 :=& g_2^{-\phi'} = g_2^{-x_0-r(\mathtt{I}x_{1,\ell}+x_3)-\frac{\gamma\phi}{\alpha}},\\
\tilde{D}_3 :=& g_2^{r+B},\\
\tilde{K}_j :=& g_2^{ry'_{1,j}+\alpha\phi'_j} = g_2^{r(y'_{1,j}+\alpha x_{1,j})+\gamma\phi_j} = g_2^{ry_{1,j}+\gamma\phi_j} \ (0 \le j \le \ell-1),\\
\tilde{K}'_j :=& g_2^{-\phi'_j} = g_2^{-rx_{1,j}-\frac{\gamma\phi_j}{\alpha}} \ (0 \le j \le \ell-1).
\end{aligned}
$$

$\mathcal{B}$ sets and returns $dk_{\mathtt{I},0} := (\tilde{R}_0^{(y)}, \tilde{R}_0^{(x)})$, $hk_{\mathtt{I},0}^{(j)} := (\tilde{R}_j^{(y)}, \tilde{R}_j^{(x)})$ $(1 \le j \le \ell-1)$, and $hk_{\mathtt{I},0}^{(\ell)} := (\tilde{D}_1, \tilde{D}'_1, \tilde{D}_2, \tilde{D}'_2, \tilde{D}_3, \{(\tilde{K}_j, \tilde{K}'_j)\}_{j=0}^{\ell-1})$.

***KI* oracle.** Without loss of generality, we fix any $j \in \{0, 1, \ldots, \ell\}$ as a special level, and suppose that $\mathcal{B}$ receives a query $(i, \mathtt{I}^*, \mathtt{time})$ such that $i \neq j$ and $T_i(\mathtt{time}) \neq T_i(\mathtt{time}^*)$ if $i < j$, where $\mathtt{I}^*$ and $\mathtt{time}^*$ are the target identity and target time, respectively. Then, $\mathcal{B}$ can generate initial semi-functional keys for $\mathtt{I}^*$ as in the *KG* oracle. Therefore, $\mathcal{B}$ can return any i-th semi-functional key for $\mathtt{I}^*$ at $\mathtt{time}$.

In the challenge phase, $\mathcal{B}$ receives $(M_0^*, M_1^*, \mathtt{I}^*, \mathtt{time}^*)$ from $\mathcal{A}$. $\mathcal{B}$ chooses $d \xleftarrow{\$} \{0,1\}$. $\mathcal{B}$ chooses $s, \mathtt{tag}^* \xleftarrow{\$} \mathbb{Z}_p$ and computes

$$
\begin{aligned}
\tilde{C}_0^* :=& M_d^* \cdot e(g_1, g_2)^{y'_0 s} e(T, g_2)^{-1},\ \tilde{C}_1^* := g_1^s,\ \tilde{C}_2^* := g_1^{\alpha s} g_1^{c_2},\\
\tilde{C}_3^* :=& \Big(\prod_{j=0}^{\ell-1} (u_{1,j}^{t_j^*}) u_{1,\ell}^{\mathtt{I}^*} w_1^{\mathtt{tag}^*} h_1\Big)^s (g_1^{c_2})^{-\sum_{j=0}^{\ell-1}(x_{1,j}t_j^*)-x_1\mathtt{I}^*-x_2\mathtt{tag}^*-x_3}.
\end{aligned}
$$

$\mathcal{B}$ sends $C^* := (\tilde{C}_0^*, \tilde{C}_1^*, \tilde{C}_2^*, \tilde{C}_3^*, \mathtt{tag}^*)$ to $\mathcal{A}$.

If $b = 0$, then the above ciphertext is semi-functional one of M_d^* by setting $\mu := c_2$. If $b = 1$, then the above ciphertext is semi-functional one of a random element of $\mathbb{G}_T$ since it holds

$$
\begin{aligned}
\tilde{C}_0^* &= M_d^* \cdot e(g_1, g_2)^{y'_0 s - x_0\mu - \eta}\\
&= M_d^* \cdot e(g_1, g_2)^{-x_0\alpha s + y_0 s - x_0\mu - \eta}\\
&= M_d^* \cdot e(g_1, g_2)^{-x_0(\alpha s+\mu)+y_0 s} e(g_1, g_2)^{-\eta}\\
&= R \cdot e(g_1, g_2)^{-x_0(\alpha s+\mu)+y_0 s},
\end{aligned}
$$

where $R = M_d^* \cdot e(g_1, g_2)^{-\eta}$.

After receiving d' from $\mathcal{A}$, $\mathcal{B}$ sends $b' = 1$ to the challenger of the DDH1 problem if $d' = d$. Otherwise, $\mathcal{B}$ sends $b' = 0$ to the challenger. □

Proof of Theorem 1. From Lemmas 1, 2, and 3, we have

$$
\begin{aligned}
Adv_{\Pi_{IKE},\mathcal{A}}^{IND\text{-}KE\text{-}CPA}(\lambda, \ell) &\le |S_{\mathsf{Real}} - S_0| + \sum_{i=1}^{q} |S_{i-1} - S_i| + |S_q - S_{\mathsf{Final}}| + |S_{\mathsf{Final}} - \frac{1}{2}|\\
&\le 4Adv_{\mathcal{G},\mathcal{B}}^{DDH1}(\lambda) + 2q \cdot Adv_{\mathcal{G},\mathcal{B}}^{DDH2}(\lambda).
\end{aligned}
$$
□

6 Chosen-Ciphertext Security

Boneh et al. [5] proposed an well-known transformation from $(\ell+1)$-level CPA-secure HIBE (and one-time signature (OTS)) to ℓ-level CCA-secure HIBE. We cannot apply this transformation to a hierarchical

IKE scheme *in a generic way* since it does not have delegating functionality. However, we can apply their techniques to the underlying Jutla-Roy HIBE of our hierarchical IKE, and therefore we obtain CCA-secure scheme. We show the detailed construction as follows. We assume a verification key vk is appropriately encoded as an element of $\mathbb{Z}_p$ when it is used in exponent of ciphertexts.

Let $\Pi_{OTS} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Ver})$ be an OTS scheme.[6] An ℓ-level hierarchical IKE scheme Π_{IKE} =(PGen, Gen, Δ-Gen, Upd, Enc, Dec) is constructed as follows.

- $\mathsf{PGen}(\lambda, \ell)$: It runs $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, g_1, g_2, e) \leftarrow \mathcal{G}$. It chooses $x_0, y_0, \{(x_{1,j}, y_{1,j})\}_{j=0}^{\ell}, \hat{x}_1, \hat{y}_1, x_2, y_2, x_3, y_3 \xleftarrow{\$} \mathbb{Z}_p$ and $\alpha \xleftarrow{\$} \mathbb{Z}_p^{\times}$, and sets

$$z = e(g_1, g_2)^{-x_0\alpha + y_0}, \quad u_{1,j} := g_1^{-x_{1,j}\alpha + y_{1,j}} \ (0 \le j \le \ell),$$
$$\hat{u}_1 := g_1^{-\hat{x}_1\alpha + \hat{y}_1}, \quad w_1 := g_1^{-x_2\alpha + y_2}, \quad h_1 := g_1^{-x_3\alpha + y_3}.$$

  It outputs

$$pp := (g_1, g_1^{\alpha}, \{u_{1,j}\}_{j=0}^{\ell}, \hat{u}_1, w_1, h_1, g_2, \{(g_2^{x_{1,j}}, g_2^{y_{1,j}})\}_{j=0}^{\ell}, g_2^{\hat{x}_1}, g_2^{\hat{y}_1}, g_2^{x_2}, g_2^{x_3}, g_2^{y_2}, g_2^{y_3}, z),$$
$$mk := (x_0, y_0).$$

- $\mathsf{Gen}(mk, ID)$: It chooses $\beta_0^{(x)}, \ldots, \beta_{\ell-1}^{(x)}, \beta_0^{(y)}, \ldots, \beta_{\ell-1}^{(y)}, r \xleftarrow{\$} \mathbb{Z}_p$, and let $B^{(j)} := \sum_{i=0}^{\ell-1} \beta_i^{(j)}$ for $j \in \{x, y\}$. It computes

$$R_j^{(y)} := g_2^{-\beta_j^{(y)}} \ (0 \le j \le \ell - 1), \quad R_j^{(x)} := g_2^{\beta_j^{(x)}} \ (0 \le j \le \ell - 1),$$
$$D_1 := (g_2^{y_2})^r, \quad D_1' := g_2^{y_0 + B^{(y)}} \left((g_2^{y_{1,\ell}})^{\mathtt{I}} g_2^{y_3} \right)^r,$$
$$D_2 := (g_2^{x_2})^{-r}, \quad D_2' := g_2^{-x_0 - B^{(x)}} \left((g_2^{x_{1,\ell}})^{\mathtt{I}} g_2^{x_3} \right)^{-r},$$
$$D_3 := g_2^r, \quad K_j := (g_2^{y_{1,j}})^r \ (0 \le j \le \ell - 1), \quad K_j' := (g_2^{x_{1,j}})^{-r} \ (0 \le j \le \ell - 1),$$
$$K_{vk} := (g_2^{\hat{y}_1})^r, \ K_{vk}' := (g_2^{\hat{x}_1})^{-r}.$$

  It outputs

$$dk_{\mathtt{I},0} := (R_0^{(y)}, R_0^{(x)}),$$
$$hk_{\mathtt{I},0}^{(i)} := (R_i^{(y)}, R_i^{(x)}) \ (1 \le i \le \ell - 1),$$
$$hk_{\mathtt{I},0}^{(\ell)} := (D_1, D_1', D_2, D_2', D_3, \{(K_j, K_j')\}_{j=0}^{\ell-1}, K_{vk}, K_{vk}').$$

- Δ-$\mathsf{Gen}(hk_{\mathtt{I},t_i}^{(i)}, \mathtt{time})$: If $t_i \neq T_i(\mathtt{time})$, it outputs $\perp$. Otherwise, parse $hk_{\mathtt{I},t_i}^{(i)}$ as $(R_i^{(y)}, R_i^{(x)}, D_1, D_1', D_2, D_2', D_3, \{(K_j, K_j')\}_{j=0}^{i-1}, K_{vk}, K_{vk}')$. It chooses $\hat{r} \leftarrow \mathbb{Z}_p$, and let $t_j := T_j(\mathtt{time})$ $(i - 1 \le j \le \ell - 1)$. It computes

$$\hat{d}_1 := D_1 (g_2^{y_2})^{\hat{r}}, \quad \hat{d}_1' := D_1' (K_{i-1})^{t_{i-1}} \left((g_2^{y_{1,\ell}})^{\mathtt{I}} \prod_{j=i-1}^{\ell-1} \left((g_2^{y_{1,j}})^{t_j} \right) g_2^{y_3} \right)^{\hat{r}},$$
$$\hat{d}_2 := D_2 (g_2^{x_2})^{-\hat{r}}, \quad \hat{d}_2' := D_2' (K_{i-1}')^{t_{i-1}} \left((g_2^{x_{1,\ell}})^{\mathtt{I}} \prod_{j=i-1}^{\ell-1} \left((g_2^{x_{1,j}})^{t_j} \right) g_2^{x_3} \right)^{-\hat{r}},$$
$$\hat{d}_3 := D_3 g_2^{\hat{r}}, \quad \hat{k}_j := K_j (g_2^{y_{1,j}})^{\hat{r}} \ (0 \le j \le i - 2), \quad \hat{k}_j' := K_j' (g_2^{x_{1,j}})^{-\hat{r}} \ (0 \le j \le i - 2),$$
$$\hat{k}_{vk} := K_{vk} (g_2^{\hat{y}_1})^{\hat{r}}, \ \hat{k}_{vk}' := K_{vk}' (g_2^{\hat{x}_1})^{\hat{r}}.$$

  It outputs $\delta_{t_{i-1}}^{(i-1)} := (\hat{d}_1, \hat{d}_1', \hat{d}_2, \hat{d}_2', \hat{d}_3, \{(\hat{k}_j, \hat{k}_j')\}_{j=0}^{i-2}, \hat{k}_{vk}, \hat{k}_{vk}')$.

[6] The formal description of the OTS is given in A.

- $\mathsf{Upd}(hk^{(i)}_{\mathtt{I},t_i}, \delta^{(i)}_{\tau_i})$: Parse $hk^{(i)}_{\mathtt{I},t_i}$ and $\delta^{(i)}_{\tau_i}$ as $(R^{(y)}_i, R^{(x)}_i, D_1, D'_1, D_2, D'_2, D_3, \{(K_j, K'_j)\}^{i-1}_{j=0}, K_{vk}, K'_{vk})$ and $(\hat{d}_1, \hat{d}'_1, \hat{d}_2, \hat{d}'_2, \hat{d}_3, \{(\hat{k}_j, \hat{k}'_j)\}^{i-1}_{j=0}, \hat{k}_{vk}, \hat{k}'_{vk})$, respectively. It outputs $hk^{(i)}_{\mathtt{I},\tau_i} := (\hat{R}^{(y)}_i, \hat{R}^{(x)}_i, \hat{D}_1, \hat{D}'_1, \hat{D}_2, \hat{D}'_2, \hat{D}_3, \{(\hat{K}_j, \hat{K}'_j)\}^{i-1}_{j=0}, \hat{K}_{vk}, \hat{K}'_{vk}) = (R^{(y)}_i, R^{(x)}_i, \hat{d}_1, \hat{d}'_1 R^{(y)}_i, \hat{d}_2, \hat{d}'_2 R^{(x)}_i, \hat{d}_3, \{\hat{k}_j, \hat{k}'_j\}^{i-1}_{j=0}, \hat{k}_{vk}, \hat{k}'_{vk})$.

- $\mathsf{Enc}(\mathtt{I}, \mathtt{time}, M)$: It first runs $(vk, sk) \leftarrow \mathsf{KGen}(\lambda)$. It chooses $s, \mathtt{tag} \xleftarrow{\$} \mathbb{Z}_p$. For $M \in \mathbb{G}_T$, it computes

$$C_0 := Mz^s,\ C_1 := g_1^s,\ C_2 := (g_1^\alpha)^s,\ C_3 := \Big(\prod_{j=0}^{\ell-1}(u_{1,j}^{t_j})u_{1,\ell}^{\mathtt{I}}\hat{u}_1^{vk}w_1^{\mathtt{tag}}h_1\Big)^s,$$

  where $t_j := T_j(\mathtt{time})$ $(0 \le j \le \ell-1)$. It also runs $\sigma \leftarrow \mathsf{Sign}(sk, (C_0, C_1, C_2, C_3, \mathtt{tag}))$, and outputs $C := (vk, C_0, C_1, C_2, C_3, \mathtt{tag}, \sigma)$.

- $\mathsf{Dec}(dk_{\mathtt{I},t_0}, \langle C, \mathtt{time}\rangle)$: If $t_0 \neq T_0(\mathtt{time})$, then it outputs $\perp$. Otherwise, parse $dk_{\mathtt{I},t_0}$ and C as $(R^{(y)}_0, R^{(x)}_0, D_1, D'_1, D_2, D'_2, D_3, K_{vk}, K'_{vk})$ and $(vk, C_0, C_1, C_2, C_3, \mathtt{tag}, \sigma)$, respectively. If $\mathsf{Ver}(vk, C_0, C_1, C_2, C_3, \mathtt{tag}, \sigma) \rightarrow 0$, then it outputs $\perp$. Otherwise, it computes

$$\hat{D}'_1 := D'_1(K_{vk})^{vk},\ \hat{D}'_2 := D'_2(K'_{vk})^{vk}.$$

  Finally, it outputs

$$M = \frac{C_0 e(C_3, D_3)}{e(C_1, D_1^{\mathtt{tag}}\hat{D}'_1)e(C_2, D_2^{\mathtt{tag}}\hat{D}'_2)}.$$

The correctness of the above IKE scheme Π_{IKE} can be checked as in our CPA-secure IKE scheme described in Section 4.

For the security of our construction above, we obtain the following theorem. The proof is omitted since this theorem can be easily proved by combining Boneh et al.'s techniques [5] and our proof techniques of Theorem 1.

Theorem 2. *If the underlying OTS scheme Π_{OTS} is sUF-OT secure and the SXDH assumption holds, then the resulting ℓ-level hierarchical IKE scheme Π_{IKE} is IND-KE-CCA secure.*

7 Public-key Encryption with Hierarchical Key Insulation

In this section, we consider the hierarchical key insulation structure in the public-key-encryption setting. Specifically, we newly formalize ℓ-level hierarchical public-key-based key-insulated encryption (PK-KIE), and propose a concrete construction for it. This proposal is the first realization of PK-KIE in the hierarchical setting.

7.1 Model and Security Definition

ℓ-level hierarchical PK-KIE takes almost the same procedure as ℓ-level hierarchical IKE. A receiver generates a public key pk and initial secret keys $dk_0, hk_0^{(1)}, \ldots, hk_0^{(\ell)}$, where dk_0 is an initial decryption key and $hk_0^{(i)}$ is an initial i-th helper key. Each helper key is stored in different devices. A sender encrypts a plaintext M with the public key pk and current time $\mathtt{time}$. The key-updating procedure is the same as that in ℓ-level hierarchical IKE. After receiving $\langle C, \mathtt{time}\rangle$, the receiver can decrypt a ciphertext C with dk_{t_0} if $t_0 = T_0(\mathtt{time})$.

An ℓ-level hierarchical PK-KIE scheme Π_{PKIE} consists of five-tuple algorithms (Setup, Δ-Gen, Upd, Enc, Dec) defined as follows.

– $(pk, dk_0, hk_0^{(1)}, \ldots, hk_0^{(\ell)}) \leftarrow \mathsf{Setup}(\lambda, \ell)$: An algorithm for key generation. It takes a security parameter λ and the maximum hierarchy depth ℓ as input, and outputs a public key pk, an initial secret key dk_0, and initial helper keys $hk_0^{(1)}, \ldots, hk_0^{(\ell)}$, where $hk_0^{(i)}$ $(1 \le i \le \ell)$ is assumed to be stored user's i-th level private device.

– $\delta^{(i-1)}_{T_{i-1}(\mathtt{time})}$ or $\perp \leftarrow \Delta\text{-}\mathsf{Gen}(hk^{(i)}_{t_i}, \mathtt{time})$: An algorithm for key update generation. It takes an i-th helper key $hk^{(i)}_{t_i}$ at a time period $t_i \in \mathcal{T}_i$ and current time $\mathtt{time}$ as input, and outputs key update $\delta^{(i-1)}_{T_{i-1}(\mathtt{time})}$ if $t_i = T_i(\mathtt{time})$; otherwise, it outputs $\perp$.

– $hk^{(i)}_{\tau_i} \leftarrow \mathsf{Upd}(hk^{(i)}_{t_i}, \delta^{(i)}_{\tau_i})$: A probabilistic algorithm for decryption key generation. It takes an i-th helper key $hk^{(i)}_{t_i}$ at a time period $t_i \in \mathcal{T}_i$ and key update $\delta^{(i)}_{\tau_i}$ at a time period $\tau \in \mathcal{T}_i$ as input, and outputs a renewal i-th helper key $hk^{(i)}_{\tau_i}$ at τ. Note that for any $t_0 \in \mathcal{T}_0$, $hk^{(0)}_{t_0}$ means dk_{t_0}.

– $\langle C, \mathtt{time}\rangle \leftarrow \mathsf{Enc}(pk, \mathtt{time}, M)$: A probabilistic algorithm for encryption. It takes a public key pk, current time $\mathtt{time}$, and a plaintext $M \in \mathcal{M}$ as input, and outputs a pair of a ciphertext and current time $\langle C, \mathtt{time}\rangle$.

– M or $\perp \leftarrow \mathsf{Dec}(dk_{t_0}, \langle C, \mathtt{time}\rangle)$: A deterministic algorithm for decryption. It takes dk_{t_0} and $\langle C, \mathtt{time}\rangle$ as input, and outputs M or $\perp$, where $\perp$ indicates decryption failure.

In the above model, we assume that Π_{PKIE} meets the following correctness property: For all security parameter λ, all $\ell := poly(\lambda)$, all $(pk, dk_0, hk^{(1)}_0, \ldots, hk^{(\ell)}_0) \leftarrow \mathsf{Setup}(\lambda, \ell)$, all $M \in \mathcal{M}$, and all $\mathtt{time} \in \mathcal{T}$, it holds that $M \leftarrow \mathsf{Dec}(dk_{T_0(\mathtt{time})}, \mathsf{Enc}(pk, \mathtt{time}, M))$, where $dk_{T_0(\mathtt{time})}$ is generated as follows: For $i = \ell, \ldots, 1$, $hk^{(i-1)}_{T_{i-1}(\mathtt{time})} \leftarrow \mathsf{Upd}(hk^{(i-1)}_{t_{i-1}}, \Delta\text{-}\mathsf{Gen}(hk^{(i)}_{T_i(\mathtt{time})}, \mathtt{time}))$, where some $t_i \in \mathcal{T}_i$ and $hk^{(0)}_{T_0(\mathtt{time})} := dk_{T_0(\mathtt{time})}$.

We consider the strong security for (hierarchical) PK-KIE, i.e., indistinguishability against key exposure and chosen ciphertext attack for PK-KIE (IND-KE-CCA). Let $\mathcal{A}$ be a PPT adversary, and $\mathcal{A}$'s advantage against IND-KE-CCA security is defined by

$$Adv^{IND\text{-}KE\text{-}CCA}_{\Pi_{PKIE},\mathcal{A}}(\lambda, \ell) := \left| \Pr\left[b' = b \;\middle|\; \begin{array}{l} (pp, dk_0, hk^{(1)}_0, \ldots, hk^{(\ell)}_0) \leftarrow \mathsf{Setup}(\lambda, \ell), \\ (M^*_0, M^*_1, \mathtt{time}^*, state) \leftarrow \mathcal{A}^{KI(\cdot,\cdot), Dec(\cdot)}(\mathsf{find}, pk), \\ b \xleftarrow{\$} \{0,1\}, C^* \leftarrow \mathsf{Enc}(pk, \mathtt{time}^*, M^*_b), \\ b' \leftarrow \mathcal{A}^{KI(\cdot,\cdot), Dec(\cdot)}(\mathsf{guess}, C^*, state) \end{array} \right] - \frac{1}{2} \right|.$$

where $KI(\cdot,\cdot)$ and $Dec(\cdot)$ are defined as follows.

KI$(\cdot,\cdot)$**:** For a query $(i, \mathtt{time}) \in \{0, 1, \ldots, \ell\} \times \mathcal{T}$, it returns $hk^{(i)}_{T_i(\mathtt{time})}$ by running $\delta^{(j-1)}_{T_{j-1}(\mathtt{time})} \leftarrow \Delta\text{-}\mathsf{Gen}(hk^{(j)}_{T_j(\mathtt{time})}, \mathtt{time})$ and $hk^{(j-1)}_{T_{j-1}(\mathtt{time})} \leftarrow \mathsf{Upd}(hk^{(j-1)}_t, \delta^{(j-1)}_{T_{j-1}(\mathtt{time})})$ for $j = \ell, \ldots, i+1$.

Dec$(\cdot)$**:** For a query $\langle C, \mathtt{time}\rangle$, it returns $\mathsf{Dec}(dk_{T_0(\mathtt{time})}, \langle C, \mathtt{time}\rangle)$.

$\mathcal{A}$ can issue any queries $(i, \mathtt{time})$ to the KI oracle if there exists at least one *special level* $j \in \{0, 1, \ldots, \ell\}$ such that

1. For any $\mathtt{time} \in \mathcal{T}$, $(j, \mathtt{time})$ is never issued to KI.
2. $(i, \mathtt{time}) \in \{0, 1, \ldots, j-1\} \times \mathcal{T}$ such that $T_i(\mathtt{time}) = T_i(\mathtt{time}^*)$ is never issued to KI.

$\mathcal{A}$ is not allowed to issue $\langle C^*, \mathtt{time}\rangle$ such that $T_0(\mathtt{time}) = T_0(\mathtt{time}^*)$ to Dec.

Definition 5 (IND-KE-CCA)**.** *An ℓ-level hierarchical PK-KIE scheme Π_{PKIE} is said to be IND-KE-CCA secure if for all PPT adversaries $\mathcal{A}$, $Adv^{IND\text{-}KE\text{-}CCA}_{\Pi_{PKIE},\mathcal{A}}(\lambda, \ell)$ is negligible in λ.*

Remark 3. *The above security definition captures strong security. In particular, the above definition is equivalent to traditonal definition of PK-KIE [2, 14] when $\ell = 1$.*

7.2 Construction

We construct an ℓ-level hierarcical PK-KIE scheme based on our CPA-secure hierarchical IKE construction and an well-known transformation from any CPA-secure IBE scheme and any OTS scheme to a CCA-secure PKE scheme [5, 8]. Therefore, this construction is similar to a CCA-secure hierarchical IKE construction proposed in Section 6. The main difference between them is that in this construction, the master key of the

Jutla-Roy HIBE scheme is used as an ℓ-th level helper key as in the existing construction of PK-KIE from an IBE scheme [2], whereas it is used as the master key in the CCA-secure hierarchical IKE construction.

An ℓ-level hierarchical PK-KIE scheme Π_{PKIE} =(Setup, Δ-Gen, Upd, Enc, Dec) is constructed as follows.

- $\mathsf{Setup}(\lambda, \ell)$: It runs $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, g_1, g_2, e) \leftarrow \mathcal{G}$. It chooses $x_0, y_0, \{(x_{1,j}, y_{1,j})\}_{j=0}^{\ell-1}, \hat{x}_1, \hat{y}_1, x_2, y_2, x_3, y_3, \xleftarrow{\$} \mathbb{Z}_p$ and $\alpha \xleftarrow{\$} \mathbb{Z}_p^\times$, and sets

$$z = e(g_1, g_2)^{-x_0\alpha + y_0}, \quad u_{1,j} := g_1^{-x_{1,j}\alpha + y_{1,j}} \ (0 \le j \le \ell - 1),$$
$$\hat{u}_1 := g_1^{-\hat{x}_1\alpha + \hat{y}_1}, \quad w_1 := g_1^{-x_2\alpha + y_2}, \quad h_1 := g_1^{-x_3\alpha + y_3}.$$

  It chooses $\beta_0^{(x)}, \ldots, \beta_{\ell-1}^{(x)}, \beta_0^{(y)}, \ldots, \beta_{\ell-1}^{(y)} \xleftarrow{\$} \mathbb{Z}_p$, computes

$$B^{(y)} := \sum_{i=0}^{\ell-1} \beta_i^{(y)}, \quad B^{(x)} := \sum_{i=0}^{\ell-1} \beta_i^{(x)}, \quad D_1' := g_2^{y_0 + B^{(y)}}, \quad D_2' := g_2^{-x_0 - B^{(x)}},$$
$$R_j^{(y)} := g_2^{-\beta_j^{(y)}} \ (0 \le j \le \ell - 1), \quad R_j^{(x)} := g_2^{\beta_j^{(x)}} \ (0 \le j \le \ell - 1).$$

  It outputs

$$pk := (g_1, g_1^\alpha, \{u_{1,j}\}_{j=0}^{\ell-1}, \hat{u}_1, w_1, h_1, g_2, \{(g_2^{x_{1,j}}, g_2^{y_{1,j}})\}_{j=0}^{\ell-1}, g_2^{x_2}, g_2^{x_3}, g_2^{y_2}, g_2^{y_3}, z),$$
$$dk_0 := (R_0^{(y)}, R_0^{(x)}) \ \ hk_0^{(i)} := (R_i^{(y)}, R_i^{(x)}) \ (1 \le i \le \ell - 1), \quad hk_0^{(\ell)} := (D_1', D_2').$$

- Δ-$\mathsf{Gen}(hk_{t_i}^{(i)}, \mathtt{time})$: If $t_i \neq T_i(\mathtt{time})$, it outputs $\perp$. Otherwise, parse $hk_{t_i}^{(i)}$ as $(R_i^{(y)}, R_i^{(x)}, D_1, D_1', D_2, D_2', D_3, \{(K_j, K_j')\}_{j=0}^{i-1}, K_{vk}, K_{vk}')$.[7] It chooses $r \leftarrow \mathbb{Z}_p$, and let $t_j := T_j(\mathtt{time})$ $(i - 1 \le j \le \ell - 1)$. It computes

$$\hat{d}_1 := D_1 (g_2^{y_2})^r, \quad \hat{d}_1' := D_1' (K_{i-1})^{t_{i-1}} \Big(\prod_{j=i-1}^{\ell-1} \big((g_2^{y_{1,j}})^{t_j} \big) g_2^{y_3} \Big)^r,$$
$$\hat{d}_2 := D_2 (g_2^{x_2})^{-r}, \quad \hat{d}_2' := D_2' (K_{i-1}')^{t_{i-1}} \Big(\prod_{j=i-1}^{\ell-1} \big((g_2^{x_{1,j}})^{t_j} \big) g_2^{x_3} \Big)^{-r},$$
$$\hat{d}_3 := D_3 g_2^r, \quad \hat{k}_j := K_j (g_2^{y_{1,j}})^r \ (0 \le j \le i - 2), \quad \hat{k}_j' := K_j' (g_2^{x_{1,j}})^{-r} \ (0 \le j \le i - 2),$$
$$\hat{k}_{vk} := K_{vk} (g_2^{\hat{y}_1})^r, \ \hat{k}_{vk}' := K_{vk}' (g_2^{\hat{x}_1})^{-r}.$$

  It outputs $\delta_{t_{i-1}}^{(i-1)} := (\hat{d}_1, \hat{d}_1', \hat{d}_2, \hat{d}_2', \hat{d}_3, \{(\hat{k}_j, \hat{k}_j')\}_{j=0}^{i-2}, \hat{k}_{vk}, \hat{k}_{vk}')$.[8]

- $\mathsf{Upd}(hk_{t_i}^{(i)}, \delta_{\tau_i}^{(i)})$: Parse $hk_{t_i}^{(i)}$ and $\delta_{\tau_i}^{(i)}$ as $(R_i^{(y)}, R_i^{(x)}, D_1, D_1', D_2, D_2', D_3, \{(K_j, K_j')\}_{j=0}^{i-1}, K_{vk}, K_{vk}')$ and $(\hat{d}_1, \hat{d}_1', \hat{d}_2, \hat{d}_2', \hat{d}_3, \{(\hat{k}_j, \hat{k}_j')\}_{j=0}^{i-1}, \hat{k}_{vk}, \hat{k}_{vk}')$, respectively. It outputs $hk_{\tau_i}^{(i)} := (\hat{R}_i^{(y)}, \hat{R}_i^{(x)}, \hat{D}_1, \hat{D}_1', \hat{D}_2, \hat{D}_2', \hat{D}_3, \{(\hat{K}_j, \hat{K}_j')\}_{j=0}^{i-1}, \hat{K}_{vk}, \hat{K}_{vk}') = (R_i^{(y)}, R_i^{(x)}, \hat{d}_1, \hat{d}_1' R_i^{(y)}, \hat{d}_2, \hat{d}_2' R_i^{(x)}, \hat{d}_3, \{\hat{k}_j, \hat{k}_j'\}_{j=0}^{i-1}, \hat{k}_{vk}, \hat{k}_{vk}')$.

- $\mathsf{Enc}(pk, \mathtt{time}, M)$: It chooses $s, \mathtt{tag} \xleftarrow{\$} \mathbb{Z}_p$, and runs $(vk, sk) \leftarrow \mathsf{KGen}(\lambda)$. For $M \in \mathbb{G}_T$, it computes

$$C_0 := M z^s, \quad C_1 := g_1^s, \quad C_2 := (g_1^\alpha)^s, \quad C_3 := \Big(\prod_{j=0}^{\ell-1} (u_{1,j}^{t_j}) \hat{u}_1^{vk} w_1^{\mathtt{tag}} h_1 \Big)^s,$$

  where $t_j := T_j(\mathtt{time})$ $(0 \le j \le \ell - 1)$. It runs $\sigma \leftarrow \mathsf{Sign}(sk, (C_0, C_1, C_2, C_3, \mathtt{tag}))$, and outputs $C := (vk, C_0, C_1, C_2, C_3, \mathtt{tag}, \sigma)$.

[7] In the case $i = \ell$, R_ℓ, D_1, D_2, D_3, and $\{(K_j, K_j')\}_{j=0}^{i-1}$ mean empty strings, and we consider these as identity elements in $\mathbb{G}_2$ when these elements are used in operations.

[8] In the case $i = 1$, $\{(\hat{k}_j, \hat{k}_j')\}_{j=0}^{\ell-1}$ means an empty string, namely we have $\delta_{t_0}^{(0)} := (\hat{d}_1, \ldots, \hat{d}_5, \hat{k}_{vk}, \hat{k}_{vk}')$.

Table 3: Parameters evaluation of our ℓ-level hierarchical PK-KIE scheme.

$\#pk$	$\#dk$	$\#hk_\ell$	$\#hk_i$
$(\ell+5)\lvert\mathbb{G}_1\rvert + (2\ell+5)\lvert\mathbb{G}_2\rvert + \lvert\mathbb{G}_T\rvert$	$7\lvert\mathbb{G}_2\rvert$	$2\lvert\mathbb{G}_2\rvert$	$(2i+9)\lvert\mathbb{G}_2\rvert$

$\#C$	Enc. Cost	Dec. Cost	Assumption
$3\lvert\mathbb{G}_1\rvert + \lvert\mathbb{G}_T\rvert + \lvert\mathbb{Z}_p\rvert$	$[0, 0, \ell+5, 1]$	$[3, 0, 2, 0]$	SXDH

$\#pk$, $\#hk^{(\ell)}$, $\#hk^{(i)}$, $\#dk$, and $\#C$ denote the sizes of public keys, ℓ-th level helper keys, i-th level helper keys ($0 \leq i \leq \ell-1$), decryption keys, and ciphertexts. k denotes the number of allowable leaked decryption keys in the scheme.

- $\mathsf{Dec}(dk_{t_0}, \langle C, \mathtt{time}\rangle)$: If $t_0 \neq T_0(\mathtt{time})$, then it outputs $\perp$. Otherwise, parse dk_{t_0} and C as $(R_0^{(y)}, R_0^{(x)}, D_1, D_1', D_2, D_2', D_3$, and $(C_0, C_1, C_2, C_3, \mathtt{tag})$, respectively. If $\mathsf{Ver}(vk, C_0, C_1, C_2, C_3, \mathtt{tag}, \sigma) \to 0$, then it outputs $\perp$. Otherwise, it computes

$$\hat{D}_1' := D_1'(K_{vk})^{vk},\ \hat{D}_2' := D_2'(K_{vk}')^{vk}.$$

  Finally, it outputs

$$M = \frac{C_0 e(C_3, D_3)}{e(C_1, D_1^{\mathtt{tag}}\hat{D}_1')e(C_2, D_2^{\mathtt{tag}}\hat{D}_2')}.$$

We can easily check the correctness in a way similar to our hierarchical IKE scheme.

For the security of the above construction, we obtain the following theorem. This theorem can be also easily proved by combining existing techniques [5, 8] and our proof techniques of Theorem 1. Therefore, we omit the proof.

Theorem 3. *If the SXDH assumption holds and Π_{OTS} is sUF-OT secure, then the resulting ℓ-level hierarchical PK-KIE scheme Π_{PKIE} is IND-KE-CCA secure.*

7.3 Parameter Evaluation and Discussion

We give a parameter evaluation of our scheme in Table 3. Again, the proposed construction is the first PK-KIE scheme in the hierarchical setting.

We compare our scheme in the non-hierarchical case with existing PK-KIE schemes. Dodis et al. [14] (strongly) CCA-secure (non-hierarchical) PK-KIE scheme under decisional Diffie-Hellman (DDH) assumption. Namely, this scheme can be realized without pairings, though it does not satisfy optimal threshold property, which means that the scheme is secure even if any polynomially many decryption keys are leaked. As a result, the number of allowable leaked decryption keys q has to be determined in the setup algorithm of their scheme, and its parameter sizes depend on q. On the other hand, our scheme satisfies the optimal threshold property, and achieves constant-size parameters when $\ell = 1$. Bellare and Palacio [2] showed a generic transformation from any CCA-secure IBE scheme to CCA-secure PK-KIE scheme. However, the resulting scheme does not meet strong security. Cheon et al. [10] showed a generic transformation from any timed-release encryption (TRE) scheme to strongly CCA-secure PK-KIE scheme. However, the resulting scheme seems less efficient than ours since the currently-known, most efficient construction of TRE scheme [**?**] needs a CPA-secure identity-based key-encapsulation system, a CCA-secure PKE scheme, and an OTS scheme, whereas our scheme is based on a specific CPA-secure IBE scheme (i.e., the Jutla-Roy IBE) and an OTS scheme.

8 Conclusion

In this paper, we first proposed an ℓ-level hierarchical key-insulated encryption without random oracles in both the identity-based and public-key setting. When $\ell = 1$, our construction achieves constant-size

parameters including public parameters, decryption and helper keys, and ciphertexts, and hence our IKE scheme is more efficient than the existing scheme [35] in the sense of parameter sizes. Our IKE scheme is based on the Jutla-Roy HIBE [24] (and its variant [27]) and techniques of threshold secret sharing schemes [4, 29]. Furthermore, we realized a hierarchical PK-KIE scheme based on our hierarchical IKE construction through the transformation techniques [5, 8].

Acknowledgments. We would like to thank anonymous PKC 2016 referees for their helpful comments. The first author was supported by Grant-in-Aid for JSPS Fellows Grant Number JP13J03998 and JP16J10532. The second author was in part supported by JSPS KAKENHI [Grant Number JP15H02710], and was in part conducted under the auspices of the MEXT Program for Promoting the Reform of National Universities.

References

[1] M. Bellare and S. Miner. A forward-secure digital signature scheme. In M. Wiener, editor, *Advances in Cryptology — CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 431–448. Springer Berlin Heidelberg, 1999.

[2] M. Bellare and A. Palacio. Protecting against key-exposure: strongly key-insulated encryption with optimal threshold. *Applicable Algebra in Engineering, Communication and Computing*, 16(6):379–396, 2006.

[3] J. Bethencourt, A. Sahai, and B. Waters. Ciphertext-policy attribute-based encryption. In *IEEE Symposium on Security and Privacy*, S&P'07, pages 321–334, May 2007.

[4] G. Blakley. Safeguarding cryptographic keys. In *Proceedings of the 1979 AFIPS National Computer Conference*, pages 313–317, Monval, NJ, USA, 1979. AFIPS Press.

[5] D. Boneh, R. Canetti, S. Halevi, and J. Katz. Chosen ciphertext security from identity based encryption. *SIAM Journal on Computing*, 36(5):1301–1328, 2007.

[6] D. Boneh, A. Sahai, and B. Waters. Functional encryption: Definitions and challenges. In Y. Ishai, editor, *Theory of Cryptography*, volume 6597 of *Lecture Notes in Computer Science*, pages 253–273. Springer Berlin Heidelberg, 2011.

[7] R. Canetti, S. Halevi, and J. Katz. A forward-secure public-key encryption scheme. In E. Biham, editor, *Advances in Cryptology — EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 255–271. Springer Berlin Heidelberg, 2003.

[8] R. Canetti, S. Halevi, and J. Katz. Chosen-ciphertext security from identity-based encryption. In C. Cachin and J. Camenisch, editors, *Advances in Cryptology - EUROCRYPT 2004*, volume 3027, pages 207–222. Springer Berlin Heidelberg, 2004.

[9] S. Chatterjee and A. Menezes. On cryptographic protocols employing asymmetric pairings — the role of Ψ revisited. *Discrete Applied Mathematics*, 159(13):1311 – 1322, 2011.

[10] J. Cheon, N. Hopper, Y. Kim, and I. Osipkov. Timed-release and key-insulated public key encryption. In G. Crescenzo and A. Rubin, editors, *Financial Cryptography and Data Security*, volume 4107, pages 191–205. Springer Berlin Heidelberg, 2006.

[11] B. den Boer. A simple and key-economical unconditional authentication scheme. *Journal of Computer Security*, 2:65–72, 1993.

[12] Y. Dodis, M. Franklin, J. Katz, A. Miyaji, and M. Yung. Intrusion-resilient public-key encryption. In M. Joye, editor, *Topics in Cryptology — CT-RSA 2003*, volume 2612 of *Lecture Notes in Computer Science*, pages 19–32. Springer Berlin Heidelberg, 2003.

[13] Y. Dodis, M. Franklin, J. Katz, A. Miyaji, and M. Yung. A generic construction for intrusion-resilient public-key encryption. In T. Okamoto, editor, *Topics in Cryptology — CT-RSA 2004*, volume 2964 of *Lecture Notes in Computer Science*, pages 81–98. Springer Berlin Heidelberg, 2004.

[14] Y. Dodis, J. Katz, S. Xu, and M. Yung. Key-insulated public key cryptosystems. In L. Knudsen, editor, *Advances in Cryptology EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 65–82. Springer Berlin Heidelberg, 2002.

[15] Y. Dodis, J. Katz, S. Xu, and M. Yung. Strong key-insulated signature schemes. In Y. Desmedt, editor, *Public Key Cryptography — PKC 2003*, volume 2567 of *Lecture Notes in Computer Science*, pages 130–144. Springer Berlin Heidelberg, 2002.

[16] Y. Dodis, W. Luo, S. Xu, and M. Yung. Key-insulated symmetric key cryptography and mitigating attacks against cryptographic cloud software. In *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security*, ASIACCS '12, pages 57–58, New York, NY, USA, 2012. ACM.

[17] S. D. Galbraith, K. G. Paterson, and N. P. Smart. Pairings for cryptographers. *Discrete Applied Mathematics*, 156(16):3113 – 3121, 2008.

[18] C. Gentry and A. Silverberg. Hierarchical ID-based cryptography. In Y. Zheng, editor, *Advances in Cryptology - ASIACRYPT 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 548–566. Springer Berlin Heidelberg, 2002.

[19] G. Hanaoka, Y. Hanaoka, and H. Imai. Parallel key-insulated public key encryption. In M. Yung, Y. Dodis, A. Kiayias, and T. Malkin, editors, *Public Key Cryptography — PKC 2006*, volume 3958 of *Lecture Notes in Computer Science*, pages 105–122. Springer Berlin Heidelberg, 2006.

[20] G. Hanaoka and J. Weng. Generic constructions of parallel key-insulated encryption. In J. Garay and R. De Prisco, editors, *Security and Cryptography for Networks*, volume 6280, pages 36–53. Springer Berlin Heidelberg, 2010.

[21] Y. Hanaoka, G. Hanaoka, J. Shikata, and H. Imai. Identity-based hierarchical strongly key-insulated encryption and its application. In B. Roy, editor, *Advances in cryptology — ASIACRYPT 2005*, volume 3788 of *Lecture Notes in Computer Science*, pages 495–514. Springer Berlin Heidelberg, 2005.

[22] G. Itkis and L. Reyzin. SiBIR: Signer-base intrusion-resilient signatures. In M. Yung, editor, *Advances in Cryptology — CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 499–514. Springer Berlin Heidelberg, 2002.

[23] T. Johansson, G. Kabatianskii, and B. Smeets. On the relation between A-codes and codes correcting independent errors. In T. Helleseth, editor, *Advances in Cryptology — EUROCRYPT'93*, volume 765 of *Lecture Notes in Computer Science*, pages 1–11. Springer Berlin Heidelberg, 1994.

[24] C. Jutla and A. Roy. Shorter quasi-adaptive NIZK proofs for linear subspaces. In K. Sako and P. Sarkar, editors, *Advances in Cryptology — ASIACRYPT 2013*, volume 8269 of *Lecture Notes in Computer Science*, pages 1–20. Springer Berlin Heidelberg, 2013.

[25] B. Libert, J.-J. Quisquater, and M. Yung. Parallel key-insulated public key encryption without random oracles. In T. Okamoto and X. Wang, editors, *Public Key Cryptography — PKC 2007*, volume 4450 of *Lecture Notes in Computer Science*, pages 298–314. Springer Berlin Heidelberg, 2007.

[26] S. Ramanna, S. Chatterjee, and P. Sarkar. Variants of Waters' dual system primitives using asymmetric pairings. In M. Fischlin, J. Buchmann, and M. Manulis, editors, *Public Key Cryptography — PKC 2012*, volume 7293 of *Lecture Notes in Computer Science*, pages 298–315. Springer Berlin Heidelberg, 2012.

[27] S. Ramanna and P. Sarkar. Efficient (anonymous) compact HIBE from standard assumptions. In S. Chow, J. Liu, L. Hui, and S. Yiu, editors, *Provable Security*, volume 8782 of *Lecture Notes in Computer Science*, pages 243–258. Springer International Publishing, 2014.

[28] A. Sahai and B. Waters. Fuzzy identity-based encryption. In R. Cramer, editor, *Advances in Cryptology —EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 457–473. Springer Berlin Heidelberg, 2005.

[29] A. Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, Nov. 1979.

[30] R. Taylor. An integrity check value algorithm for stream ciphers. In D. Stinson, editor, *Advances in Cryptology — CRYPTO'93*, volume 773 of *Lecture Notes in Computer Science*, pages 40–48. Springer Berlin Heidelberg, 1994.

[31] Y. Watanabe and J. Shikata. Identity-based hierarchical key-insulated encryption without random oracles. In C.-M. Cheng, K.-M. Chung, G. Persiano, and B.-Y. Yang, editors, *Public-Key Cryptography – PKC 2016, Part I*, volume 9614 of *Lecture Notes in Computer Science*, pages 255–279, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg.

[32] B. Waters. Efficient identity-based encryption without random oracles. In R. Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494, pages 114–127. Springer Berlin Heidelberg, 2005.

[33] B. Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In S. Halevi, editor, *Advances in Cryptology — CRYPTO 2009*, volume 5677, pages 619–636. Springer Berlin Heidelberg, 2009.

[34] J. Weng, S. Liu, K. Chen, and C. Ma. Identity-based parallel key-insulated encryption without random oracles: Security notions and construction. In R. Barua and T. Lange, editors, *Progress in Cryptology - INDOCRYPT 2006*, volume 4329, pages 409–423. Springer Berlin Heidelberg, 2006.

[35] J. Weng, S. Liu, K. Chen, D. Zheng, and W. Qiu. Identity-based threshold key-insulated encryption without random oracles. In T. Malkin, editor, *Topics in Cryptology – CT-RSA 2008*, volume 4964, pages 203–220. Springer Berlin Heidelberg, 2008.

A Definitions

We give the formal definitions of the CBDH and DBDH assumptions and OTS. In the following, we assume the Type-1 pairing (i.e., $\mathbb{G} := \mathbb{G}_1 = \mathbb{G}_2$).

Computational Bilinear Diffie–Hellman (CBDH) Assumption. Let $\mathcal{A}$ be a PPT adversary and we consider $\mathcal{A}$'s advantage against the CBDH problem as follows.

$$Adv^{CBDH}_{\mathcal{G},\mathcal{A}}(\lambda) := \Pr\left[\begin{array}{l|l} T = e(g,g)^{c_1c_2c_3} & \begin{array}{l}(p,\mathbb{G},\mathbb{G}_T,g,e) \leftarrow \mathcal{G},\\ c_1,c_2,c_3 \xleftarrow{\$} \mathbb{Z}_p,\\ T \leftarrow \mathcal{A}(\lambda,g,g^{c_1},g^{c_2},g^{c_3})\end{array}\end{array}\right].$$

Definition 6. *The CBDH assumption relative to a generator $\mathcal{G}$ holds if for all PPT adversaries $\mathcal{A}$, $Adv^{CBDH}_{\mathcal{G},\mathcal{A}}(\lambda)$ is negligible in λ.*

Decisional Bilinear Diffie–Hellman (DBDH) Assumption. Let $\mathcal{A}$ be a PPT adversary and we consider $\mathcal{A}$'s advantage against the DBDH problem as follows.

$$Adv^{DBDH}_{\mathcal{G},\mathcal{A}}(\lambda) := \left|\Pr\left[\begin{array}{l|l} b' = b & \begin{array}{l}(p,\mathbb{G},\mathbb{G}_T,g,e) \leftarrow \mathcal{G},\\ c_1,c_2,c_3 \xleftarrow{\$} \mathbb{Z}_p,\\ b \xleftarrow{\$} \{0,1\},\\ \text{if } b=1 \text{ then } T := \hat{e}(g,g)^{c_1c_2c_3},\\ \text{else } T \xleftarrow{\$} \mathbb{G}_T,\\ b' \leftarrow \mathcal{A}(\lambda,g,g^{c_1},g^{c_2},g^{c_3},T)\end{array}\end{array}\right] - \frac{1}{2}\right|.$$

Definition 7. *The DBDH assumption relative to a generator $\mathcal{G}$ holds if for all PPT adversaries $\mathcal{A}$, $Adv^{DBDH}_{\mathcal{G},\mathcal{A}}(\lambda)$ is negligible in λ.*

One-time signature. An OTS scheme Π_{OTS} consists of three-tuple algorithms (KGen, Sign, Ver) defined as follows.

- $(vk, sk) \leftarrow \mathsf{KGen}(\lambda)$: It takes a security parameter λ and outputs a pair of a public key and a secret key (vk, sk).
- $\sigma \leftarrow \mathsf{Sign}(sk, m)$: It takes the secret key sk and a message $m \in \mathcal{M}$ and outputs a signature σ.
- 1 or $0 \leftarrow \mathsf{Ver}(vk, m, \sigma)$: It takes the public key vk and a pair of a message and a signature (m, σ), and then outputs 1 or 0.

We assume that Π_{OTS} meets the following *correctness* property: For all $\lambda \in \mathbb{N}$, all $(vk, sk) \leftarrow \mathsf{KGen}(\lambda)$, and all $m \in \mathcal{M}$, it holds that $1 \leftarrow \mathsf{Ver}(vk, (m, \mathsf{Sign}(sk, m)))$.

We describe the notion of strong unforgeability against one-time attack ($\mathsf{sUF\text{-}OT}$). Let $\mathcal{A}$ be a PPT adversary, and $\mathcal{A}$'s advantage against sUF-OT security is defined by

$$Adv^{sUF\text{-}OT}_{\Pi_{OTS},\mathcal{A}}(\lambda) := \Pr\left[\begin{array}{l|l} 1 \leftarrow \mathsf{Ver}(vk, m^*, \sigma^*) \wedge (m^*, \sigma^*) \neq (m, \sigma) & \begin{array}{l}(vk, sk) \leftarrow \mathsf{KGen}(\lambda),\\ (m^*, \sigma^*) \leftarrow \mathcal{A}^{Sign(\cdot)}(vk)\end{array}\end{array}\right].$$

$Sign(\cdot)$ is a *signing oracle* which takes a message m as input, and then returns σ by running $\mathsf{Sign}(sk, m)$. $\mathcal{A}$ is allowed to access to the above oracle only once.

Definition 8. *An OTS scheme Π_{OTS} is said to be sUF-OT secure if for all PPT adversaries $\mathcal{A}$, $Adv^{sUF\text{-}OT}_{\Pi_{OTS},\mathcal{A}}(\lambda)$ is negligible in λ.*