More Efficient Adaptively Secure Revocable Hierarchical Identity-based Encryption with Compact Ciphertexts: Achieving Shorter Keys and Tighter Reductions

Atsushi Takayasu*

April 23, 2021

Abstract

Revocable hierarchical identity-based encryption (RHIBE) is a variant of the standard hierarchical identity-based encryption (HIBE) satisfying the key revocation functionality. Recently, the first adaptively secure RHIBE scheme with compact ciphertexts was proposed by Emura et al. by sacrificing the efficiency of the schemes for achieving adaptive security so that the secret keys are much larger than Seo and Emura’s selectively secure scheme with compact ciphertexts. In this paper, we propose a more efficient adaptively secure RHIBE scheme with compact ciphertexts. Our scheme has much shorter secret keys and key updates than Emura et al.’s scheme. Moreover, our scheme has much shorter key updates than Seo and Emura’s selectively secure scheme. Emura et al. proved the adaptive security of their scheme by reducing the security of the underlying HIBE schemes to that of their proposed RHIBE scheme, where the adaptive security of the HIBE scheme is inherently proven through the dual system encryption methodology. In contrast, we prove the adaptive security of the proposed RHIBE scheme directly through the dual system encryption methodology. Furthermore, our security proof achieves a tighter reduction than that of Emura et al.

*National Institute of Information and Communications Technology (NICT), Japan. takayasu@nict.go.jp
# Contents

1 Introduction 3  
1.1 Background 3  
1.2 Our Contribution 3  
1.3 Technical Overview 4  
1.4 Related Work 5  
1.5 Roadmap 6  

2 Preliminaries 6  
2.1 Bilinear Groups 6  
2.2 RHIBE 7  

3 Proposed RHIBE Scheme 10  
3.1 CS Method 10  
3.2 Construction 10  
3.3 Correctness 15  

4 Main Theorem 17  
4.1 Auxiliary Distributions 18  
4.2 Proof of Main Theorem 19  

5 Adaptive Security against the Type-II Adversary 20  
5.1 Proof of Lemma 2 25  

6 Adaptive Security against the Type-I Adversary 40  
6.1 Proof of Lemma 1 46  

7 Comparison 57  
7.1 Comparison among RHIBE Schemes with Compact Ciphertexts 58  
7.2 Comparison among RHIBE Schemes with Adaptive Security 58  

8 Conclusion 60
1 Introduction

1.1 Background

Identity-based encryption (IBE) [Sha84] is an extension of the traditional public key encryption. We can use an arbitrary string $ID$ as the public key of IBE. The key generation center (KGC) of IBE takes the master public key $MPK$ and $ID$ as input and computes a secret key $sk_{ID}$. Hierarchical IBE (HIBE) is an extension of IBE. In HIBE, a vector of arbitrary strings $ID = (id_1, \ldots, id_\ell)$ can be used as the public key of HIBE. In an HIBE system, not only the KGC, but also the user $ID'$ with a secret key $sk_{ID'}$ can create a secret key $sk_{ID'}$ iff $ID'$ is a prefix of $ID$. So far, several efficient and adaptively secure HIBE schemes have been proposed over prime-order pairing groups (e.g., [BKP14, CGW15, CG17, CW14, GCTC16, LP19, LP20a, LP20b, Lew12, OT15, RS14, Wat09]) through Waters’ dual system encryption methodology [Wat09].

Despite the convenience of using an HIBE system, such systems do not have a naive way to revoke malicious users dynamically and efficiently. Boldyreva et al. [BGK08] resolved this problem by introducing revocable IBE (RIBE), a variant of IBE with a scalable revocation functionality. They proposed the first RIBE scheme by utilizing a subset cover framework [NNL01] such as the complete subtree (CS) method. Then, Seo and Emura [SE13b] refined the security model of RIBE by introducing a new security notion called decryption key exposure resistance (DKER). Later, Seo and Emura [SE13a] introduced revocable HIBE (RHIBE). Seo and Emura [SE15] and Katsumata et al. [KMT19] refined the security model by introducing the DKER and an insider security as the security requirements of RHIBE.

Although there are several adaptively secure RIBE schemes over prime-order pairing groups under the standard assumptions [LV09, ML19, SE13b, TW21], the first hierarchical analog was recently proposed by Emura et al. [ETW20]. Emura et al.’s scheme achieves compact ciphertexts or compact master public keys. Specifically, Emura et al. introduced several algebraic properties of pairing-based HIBE schemes and proposed a semi-generic construction of RHIBE from pairing-based HIBE, e.g., [CG17, CW14, GCTC16]. Thus, they used those HIBE schemes whose adaptive security was proved through the dual system encryption methodology as a building block and constructed adaptively secure RHIBE schemes. To achieve adaptive security, Emura et al. sacrificed the efficiency; their proposed RHIBE schemes have much larger secret keys than the existing selectively secure RHIBE schemes. Therefore, constructing more efficient RHIBE schemes with adaptive security is an interesting research topic. Recently, other adaptively secure RHIBE schemes have been proposed by Lee and Kim [LK21] and Emura et al. [ETW21] although both schemes cannot achieve compact ciphertexts.

1.2 Our Contribution

In this paper, we propose a more efficient adaptively secure RHIBE scheme with compact ciphertexts. Our RHIBE scheme is a modification of Chen and Gong’s HIBE scheme with compact ciphertexts [CG17] that satisfies adaptive security under the standard $k$-linear assumption. We followed the design principle of Lee and Park’s selectively secure RHIBE scheme [LP18] and constructed the proposed RHIBE scheme. Our proposed RHIBE scheme has much shorter secret keys and key updates than those of Emura et al.’s RHIBE scheme [ETW20], which was constructed from the same Chen and Gong’s HIBE scheme. Moreover, our proposed RHIBE scheme has much shorter key update than Seo and Emura’s selectively secure scheme with compact ciphertexts [SE15].
1.3 Technical Overview

We provide a brief overview of our proof technique. Similar to the schemes of Emura et al. and Lee and Park, the master secret key $k$ of our scheme is split into ID’s secret key $sk_{ID}$ and the parent user $pa(ID)$’s key update $ku_{pa(ID),T}$ at time period $T$ as two-out-of-two secret sharing. More concretely, $sk_{ID}$ contains several sub-secret keys $sk_{ID,\theta}$, and $ku_{pa(ID),T}$ contains several sub-key updates $ku_{pa(ID),T,\theta}$ that are associated with nodes $\theta$ in a binary tree $BT_{pa(ID)}$ managed by $pa(ID)$. Specifically, $sk_{ID,\theta}$ and $ku_{pa(ID),T}$ are HIBE secret keys with $k_{pa(ID),\theta}$ and $k - k_{pa(ID),\theta}$ as the master secret key-parts, respectively, where $k_{pa(ID),\theta}$ is the uniformly random element to mask the master secret key $k$. Given the key update $ku_{pa(ID),T}$, a user ID can decrypt ciphertext $ct_{ID,T}$ in the same time period $T$ if there are $sk_{ID,\theta}$ and $ku_{pa(ID),T}$ that share the same node $\theta$. In other words, $sk_{ID,\theta}$ and $ku_{pa(ID),T}$ associated with the same node $\theta$ can delete the random mask $k_{pa(ID),\theta}$ and exploit the true master secret key $k$.

Overview of Emura et al.’s Proof. Emura et al. [ETW20] proved the adaptive security by simply extending the technique of the selectively secure RHIBE scheme. First, they introduced the adaptive node division technique that divides all nodes $\theta$ in the security proof into exclusive two groups. Let $(ID^*,T^*)$ denote the tuple of the challenge identity and challenge time period. Emura et al.’s adaptive node division technique ensures that all $sk_{ID,\theta}$ whose nodes are members of the first group satisfy $ID \notin prefix^+(ID^*)$, whereas all $ku_{pa(ID),T,\theta}$ whose nodes are members of the second group satisfy $pa(ID) \notin prefix^+(ID^*) \lor T \neq T^*$. Then, Emura et al. switched the positions of the master secret key $k$ so that $sk_{ID,\theta}$ and $ku_{pa(ID),T,\theta}$ associated with the node $\theta$ in the first group are HIBE secret keys with $k - k_{pa(ID),\theta}$ and $k_{pa(ID),\theta}$, respectively, as the master secret key-parts. Therefore, the reduction algorithm itself can create all $sk_{ID,\theta}$ and $ku_{pa(ID),T,\theta}$ associated with the node $\theta$ in the second and first groups, respectively, since the master secret key $k$ is not required. Moreover, the reduction algorithm could interact with the HIBE challenger and receive $sk_{ID,\theta}$ and $ku_{pa(ID),T,\theta}$ associated with the node $\theta$ in the first and second groups based on the conditions $ID \notin prefix^+(ID^*)$ and $pa(ID) \notin prefix^+(ID^*) \lor T \neq T^*$, respectively.

Here, the one problem to avoid is that the adversary can receive not only the decryption-purpose secret keys $sk_{ID,\theta}$, but also the delegation-purpose secret key $delk_{pa(ID),\theta}$. In short, setting $delk_{pa(ID),\theta} = k_{pa(ID),\theta}$ as the delegation-purpose secret keys is sufficient for achieving correctness; however, it means that the adversary can receive $k_{pa(ID),\theta}$. In this case, we cannot switch the positions of the master secret key $k$ since the reduction algorithm cannot answer $delk_{pa(ID),\theta} = k - k_{pa(ID),\theta}$ in the first group. Therefore, Emura et al. set the delegation-purpose secret keys $delk_{pa(ID),\theta}$ as the HIBE secret keys with $k_{pa(ID),\theta}$ as the master secret key-part. As a result, even when the delegation-purpose secret key $delk_{pa(ID),\theta}$ becomes the HIBE secret key with $k - k_{pa(ID),\theta}$ after switching the master secret key, the reduction algorithm can interact with the HIBE challenger and receive the corresponding HIBE secret keys $delk_{pa(ID),\theta}$ owing to the condition that $pa(ID) \notin prefix^+(ID^*)$. In contrast, the modification to answer $delk_{pa(ID),T}$ results in a larger secret key.

Overview of Our Proof. Although Emura et al. proved the adaptive security by reducing the security of the underlying HIBE scheme to the security of their proposed RHIBE scheme, we prove the adaptive security of our proposed RHIBE scheme directly by Waters’ dual system encryption methodology [Wat09] and its variants [CGW15, CG17, CW14]. To prove the security of HIBE through the dual system encryption methodology, we use semi-functional distributions for the challenge ciphertexts and secret keys in addition to the normal distributions in the real scheme. In brief, the semi-functional secret keys are HIBE secret keys for the same identity with $k + \alpha a^\perp$ as the master secret key-part, where $a^\perp$ is a specific vector, and $\alpha$ is a uniformly random element in $\mathbb{Z}_p$. The normal secret keys can decrypt both normal and semi-functional ciphertexts. Although the semi-functional secret keys can decrypt normal ciphertexts, they cannot decrypt...
semifunctional ciphertexts. In the proof, we first change the challenge ciphertexts from normal to semi-functional. Then, we change each secret key queried by the adversary from normal to semi-functional one by one. Once all the secret keys $sk_{1D}$ are changed to the semi-functional type, $\alpha a^\perp$ masks the distribution of the master secret key $k$; then, the plaintext of the semi-functional challenge ciphertext is information theoretically hidden.

Unlike in Emura et al.’s proof, we do not switch the positions of the master secret key $k$ so that we set the delegation-purpose secret keys $delk_{pa(ID), \theta} = k_{pa(ID), \theta}$ as the compact form. In turn, we change the position of the semi-functional randomness; this process is called a semi-functional randomness switching that was implicitly introduced by Takayasu and Watanabe [TW21]. When $pa(ID) \notin \text{prefix}^+(ID^*)$, we change all $ku_{pa(ID), T, \theta}$ to be semi-functional through the standard dual system argument. To prove $ku_{pa(ID), T, \theta}$ such that $pa(ID) \in \text{prefix}^+(ID^*)$, we employ the semi-functional randomness switching. Here, we provide an overview of the simplest form of the semi-functional randomness switching. If the adversary does not receive both the parent user $pa(ID)$’s secret key $sk_{pa(ID)}$ and ID’s secret key $sk_{1D}$ such that ID $\in \text{prefix}^+(ID^*)$, the reduction algorithm changes all secret keys $sk_{1D}$ from normal to be semi-functional through the standard dual system argument. Specifically, $sk_{1D}$ becomes the HIBE secret keys with $k_{pa(ID), \theta} + \alpha_{ID, \theta}a^\perp$ as the master secret key-parts, where $\alpha_{ID, \theta}$ is the uniformly random element in $\mathbb{Z}_p$. Once all the secret keys $sk_{1D}$ are changed to be semi-functional, $sk_{1D}$ and $ku_{pa(ID), T, \theta}$ are the HIBE secret keys with $k_{pa(ID), \theta} + \alpha_{ID, \theta}a^\perp$ and $k - k_{pa(ID), \theta}$, respectively, as the master secret key-parts. Note that the adversary does not receive $delk_{pa(ID), \theta} = k_{pa(ID), \theta}$. Observe that $k_{pa(ID), \theta} + \alpha a^\perp$ is the uniformly random element; thus, if we set $delk_{pa(ID), \theta} = k_{pa(ID), \theta} + \alpha a^\perp$, $delk_{pa(ID), \theta}$ is properly distributed. Furthermore, $sk_{1D}$ and $ku_{pa(ID), T, \theta}$ become HIBE secret keys with $delk_{pa(ID), \theta} + (\alpha_{ID, \theta} - \alpha)a^\perp$ and $k + \alpha a^\perp - delk_{pa(ID), \theta}$, respectively, as the master secret key-parts. Here, $\alpha_{ID, \theta} - \alpha$ is a properly distributed uniformly random element in $\mathbb{Z}_p$. Thus, we successfully switch the positions of the semi-functional randomness $\alpha a^\perp$ from $sk_{pa(ID), \theta}$ to $ku_{pa(ID), T, \theta}$ by using $delk_{pa(ID), \theta}$ as the bridge. By using semi-functional randomness switching, we can change all required keys to be semi-functional and successfully prove the adaptive security of the proposed RHIBE scheme.

1.4 Related Work

Boneh and Franklin [BF01] pointed out the necessity of the revocation functionality for IBE. Boldyreva et al. [BGK08] introduced the concept of RIBE for achieving the scalable revocation and proposed the first RIBE scheme with selective security. The first adaptively secure RIBE scheme was proposed by Libert and Vergnaud [LV09]. Seo and Emura [SE13b] introduced a new security notion for RIBE called DKER and proposed the first RIBE scheme with DKER. All these schemes are constructed over pairing groups. Subsequently, several adaptively secure RIBE schemes with DKER were proposed over pairing groups [ISW17, Lee19, LLP17, WLXZ14, WES17], improving the efficiency and/or security. Then, RIBE schemes from the LWE assumption [CLL+12], the CDH assumption without pairing and the factoring assumption of Blum integers [HLCL18], and the code-based assumption [CCKS18] were proposed though they did not satisfy DKER. To break the varier of DKER without pairing, Takayasu and Watanabe [TW17] proposed a lattice-based RIBE scheme with bounded DKER. Their scheme, unlike other known RIBE schemes with DKER, satisfies the anonymity. Takayasu and Watanabe [TW21] also constructed a pairing-based anonymous RIBE scheme with bounded DKER. Katsumata et al. [KMT19] proposed the generic construction of RIBE with DKER by combining RIBE without DKER and 2-level HIBE. The result implies that RIBE without DKER implies RIBE with DKER based on [DG17]. Ma and Lin [ML19] proposed the generic construction of RIBE with DKER from 2-level HIBE.

RHIBE was first introduced by Seo and Emura [SE13a]. Unfortunately, it does not have a
convincing security definition since the adversary cannot receive the delegation-purpose secret keys 
\(\text{del}_{\text{pa}(\text{ID}),\theta}\) of corrupted parent users \(\text{pa}(\text{ID})\). Seo and Emura [SE15] refined the security definition
to resolve the above issue by introducing a new security notion called insider security; encryption
schemes are regarded as RHIBE only when they satisfy insider security. Furthermore, they also
defined DKER for RHIBE. In the security model, several RHIBE schemes were proposed over
pairing-groups [ESY16, LP18, RLPL15, SE15]. Katsumata et al. [KMT19] further refined the
security model and introduced a stronger notion of DKER. Katsumata et al. proposed a lattice-
based RHIBE scheme, and Wang et al. [WZH+19] proposed a more efficient variant. None of these
RHIBE schemes in the standard model satisfy adaptive security. Furthermore, most pairing-based
RHIBE schemes [LP18, RLPL15, SE15] are based on nonstandard \(q\)-type assumptions. Emura et
al. [ETW20] proposed the first adaptively secure RHIBE schemes in the standard model. They
introduced several algebraic properties of known pairing-based HIBE schemes and proposed the
generic construction of RHIBE from pairing-based HIBE. Thus, the instantiations capture the
adaptively secures RHIBE schemes under the standard \(k\)-linear assumption. Recently, Lee and
Kim [LK21] and Emura et al. [ETW21] proposed a generic construction of RHIBE from HIBE.
These schemes inherently suffer from large ciphertexts.

1.5 Roadmap

In Section 2, we review the pairing groups and the definition of RHIBE. In Section 3, we propose
our RHIBE scheme. In Section 4, we provide the main security theorem and its high level proof. In
Sections 5 and 6, we prove the core lemmata for proving the main security theorem. In Section 7,
we compare our proposed RHIBE scheme with the other known RHIBE schemes.

2 Preliminaries

For two non-negative integers \(a\) and \(b\) such that \(a \leq b\), let \([a, b] := \{a, a + 1, \ldots, b\}\) and \([a] := [1, a]\).
Let a lowercase bold letter \(a\) and an uppercase bold letter \(A\) denote a column vector and matrix,
respectively. Throughout the paper, let \(\lambda\) denote the security parameter. For a finite set \(S\), let
\(x \leftarrow_R S\) denote sampling \(x\) from \(S\) uniformly at random. For two probability distributions \(P\) and \(Q\)
with a support \(S\), let \(\frac{1}{2} \sum_{x \in S} |P(x) - Q(x)|\) denote the statistical distance. For two security games
\(\text{Game}_A\) and \(\text{Game}_B\), let \(\text{Game}_A \approx_c \text{Game}_B\) denote that \(\text{Game}_A\) and \(\text{Game}_B\) are computationally
indistinguishable from an adversary’s view and let \(\text{Game}_A \equiv \text{Game}_B\) denote that \(\text{Game}_A\) and \(\text{Game}_B\)
are identically distributed from an adversary’s view. We use the same notation \(\approx_c\) and \(\equiv\) for two
probability distributions.

2.1 Bilinear Groups

Let \(G\) denote a prime-order pairing groups generator. Given the security parameter \(1^\lambda\) as input, \(G\)
outputs \((p, G_1, G_2, G_T, g_1, g_2, e)\), where \(p\) is a \(\Theta(\lambda)\)-bit prime number, \(G_1, G_2, G_T\) are cyclic groups
of order \(p\), \(g_1\) and \(g_2\) are the generators of \(G_1\) and \(G_2\), respectively, and \(e : G_1 \times G_2 \to G_T\) is an
efficiently computable non-degenerate bilinear map. Let \([a]_1 := g_1^a \in G_1\) denote a group element,
where \(a \in \mathbb{Z}_p\). Similarly, let \([a]_1\) and \([A]_1\) denote a vector and matrix of group elements. We use
the same notations for the other groups \(G_2\) and \(G_T\). For two matrices \(A \in \mathbb{Z}_p^{\ell \times m}\) and \(B \in \mathbb{Z}_p^{\ell \times n}\),
let \(e([A]_1, [B]_2) = [A^\top B]_T\).

Next, we review the matrix decisional Diffie-Hellman (MDDH) assumption [EHK+17].
Definition 1 (Matrix Distribution). For a positive integer $k$, a matrix distribution $\mathcal{D}_k$ outputs a rank $k$ matrix $A \in \mathbb{Z}_p^{(k+1) \times k}$ and non-zero vector $a^\top \in \mathbb{Z}_p^{k+1}$ satisfying $A^\top a^\top = 0$.

Without loss of generality, we assume that the top $k \times k$ sub-matrix of $A$ output by $\mathcal{D}_k$ is full-rank. Briefly speaking, for $A \leftarrow \mathcal{D}_k$ the MDDH assumption states that $(|A|_1, |A|_2) \approx (|A|_1, |u|_1)$ for uniformly random vectors $s \leftarrow_R \mathbb{Z}_p^k$ and $u \leftarrow_R \mathbb{Z}_p^{k+1}$.

Definition 2 (MDDH Assumption in $\mathbb{G}_1$). Let $(p, G_1, G_2, G_T, g_1, g_2, e) \leftarrow G(1^\lambda)$ denote a description of a pairing group. The MDDH assumption in $\mathbb{G}_1$ states that the advantage function

$$\text{Adv}_{\mathcal{A}}^{\text{MDDH-}G_1}(\lambda) := \left| \Pr \left[ \mathcal{A}(G(1^\lambda), |A|_1, |A|_2) = 1 \right] - \Pr \left[ \mathcal{A}(G(1^\lambda), |A|_1, |u|_1) = 1 \right] \right|$$

is negligible in $\lambda$ for all PPT adversary $\mathcal{A}$, where $A \leftarrow \mathcal{D}_k$, $s \leftarrow_R \mathbb{Z}_p^k$, $u \leftarrow_R \mathbb{Z}_p^{k+1}$.

We also define the MDDH assumption in $\mathbb{G}_2$ in the same way. The $k$-linear assumption is a particular case of the MDDH assumption when the $k \times k$ sub-matrix of $A$ is a diagonal matrix with $a_i \leftarrow_R \mathbb{Z}_p^*$ in $i$-th diagonal and the bottom row vector of $A$ is $(1, 1, \ldots, 1)$. In this case, we can set $a^\top = (a_1^{-1}, \ldots, a_k^{-1}, -1)^\top$. The symmetric external Diffie-Hellman (SXDH) assumption is a particular case of the $k$-linear assumption for $k = 1$.

2.2 RHIBE

In this section, we review the definition for RHIBE by following [KMT19].

Hierarchical Identities. Let $I$ denote an identity space and let $id \in I$ denote an element identity. Let $ID = (id_1, \ldots, id_\ell)$ denote an identity that is a vector of element identities and let $|ID| := \ell$ denote the length of the identity. For $ID = (id_1, \ldots, id_{|ID|})$, let $pa(ID) := (id_1, \ldots, id_{|ID|} - 1 = id_{|ID|})$ denote a parent of $ID$ and let $ID[\ell] := (id_1, \ldots, id_\ell)$ denote a length $\ell$ prefix of $ID$ for $\ell \leq |ID|$. Let $\text{prefix}^+(ID) := \{ID[1], ID[2], \ldots, ID[|ID|] = \emptyset\}$ denote a set of identities that are prefix of $ID$ and $ID$ itself.

Syntax. An RHIBE scheme $\Pi$ consists of six algorithms (Setup, Enc, GenSK, KeyUp, GenDK, Dec) defined as follows.

- $\text{Setup}(1^\lambda, L) \rightarrow (\text{MPK}, \text{sk}_{\text{kgc}})$: The setup algorithm takes security parameter $1^\lambda$ and the maximum depth of the hierarchy $L \in \mathbb{N}$ as input, and outputs a master public key MPK and the KGC’s secret key $\text{sk}_{\text{kgc}}$.
- $\text{Enc}($MPK, ID, T, M$) \rightarrow \text{ct}_{\text{ID},T}$: The encryption algorithm takes MPK, an identity $ID \in I^{\text{ID}}$, time period $T \in T$, and a plaintext $M \in \mathcal{M}$ as input, and outputs a ciphertext $\text{ct}_{\text{ID},T}$.
- $\text{GenSK}($MPK, $\text{sk}_{\text{pa(ID)}}, ID$$) \rightarrow (\text{sk}_{ID}, \text{sk}'_{\text{pa(ID)}})$: The secret key generation algorithm takes MPK, a parent’s secret key $\text{sk}_{\text{pa(ID)}}$, and an identity $ID \in I^{\text{ID}}$ as input, and outputs $\text{sk}_{ID}$ for $ID$ and the “updated” $\text{sk}'_{\text{pa(ID)}}$.
- $\text{KeyUp}($MPK, T, $\text{sk}_{ID}, \text{RL}_{ID,T}, \text{ku}_{\text{pa(ID)},T}$$) \rightarrow (\text{ku}_{ID,T}, \text{sk}'_{\text{ID}})$: The key update information generation algorithm takes MPK, $T \in T$, $\text{sk}_{ID}$ for $ID \in I^{\text{ID}}$, revocation list $\text{RL}_{ID,T} \subseteq I_{\text{ID}}$, and a parent’s key update $\text{ku}_{\text{pa(ID)},T}$ as input, and outputs $\text{ku}_{ID,T}$ and the “updated” $\text{sk}'_{\text{ID}}$. As a special case, we define $\text{ku}_{\text{pa(kgc)},T} := \perp$ for all $T \in T$.
- $\text{GenDK}($MPK, $\text{sk}_{ID}, \text{ku}_{\text{pa(ID)}}, T$$) \rightarrow \text{dk}_{ID,T}$ or $\perp$: The decryption key generation algorithm, which takes MPK, $\text{sk}_{ID}$ for $ID \in I^{\text{ID}}$, and $\text{ku}_{\text{pa(ID)},T}$ as input, and outputs a decryption key $\text{dk}_{ID,T}$ for $T \in T$ or the special symbol $\perp$, indicating that $ID$ or some of its ancestors have been revoked.
• Dec(MPK, dk_{ID,T}, ct_{ID,T}) → M: The decryption algorithm takes MPK, dk_{ID,T}, and ct_{ID,T} as input, and outputs the decryption result M.

Correctness. We require ciphertext ct_{ID,T} to be decrypted properly by a correctly-generated decryption key dk_{ID,T} for the same ID and T when ID is not revoked at T. In other words, for all \( \lambda \in \mathbb{N}, L \in \mathbb{N}, \) \((PP, sk_{kgc}) \leftarrow \text{Setup}(1^\lambda, L), \ell \in [L], \) \( \text{ID} \in (\mathcal{I})^\ell, \) \( T \in \mathcal{T}, \) \( M \in \mathcal{M}, \) \( RL_{kgc,T} \subseteq \mathcal{I}, \) \( RL_{ID,T} \subseteq \mathcal{I}^{|ID|}, \ldots, RL_{ID_{\ell-1},T} \subseteq \mathcal{I}^{|ID_{\ell-1}|}, \) if \( ID' \notin RL_{pa(ID'),T} \) holds for all \( ID' \in \text{prefix}^+(ID) \). Then, we require \( M' = M \) to hold after executing the following procedures.

1. \((ku_{kgc,T}, sk_{kgc}) \leftarrow \text{KeyUp}(PP, T, sk_{kgc}, RL_{kgc,T}, \bot).\)
2. For all \( ID' \in \text{prefix}^+(ID) \) (in short-to-long order), execute the following (2.1) and (2.2):
   1. \((sk_{ID'}, sk_{pa(ID')}) \leftarrow \text{GenSK}(PP, sk_{pa(ID')}, ID').\)
   2. \((ku_{ID',T}, sk_{ID'}) \leftarrow \text{KeyUp}(PP, T, sk_{ID'}, RL_{ID',T}, ku_{pa(ID'),T}).\)
3. \(dk_{ID,T} \leftarrow \text{GenDK}(PP, sk_{ID}, ku_{pa(ID),T}).\)
4. \(ct \leftarrow \text{Enc}(PP, ID, T, M).\)
5. \(M' \leftarrow \text{Dec}(PP, dk_{ID,T}, ct).\)

Security Definition. Let \( \Pi \) be an RHIBE scheme. Adaptive security of RHIBE is defined by a security game between adversary \( A \) and challenger \( C. \) The game is parameterized by security parameter \( \lambda \) and polynomial \( L = L(\lambda) \) representing the maximum hierarchical depth. Let a global counter \( T_{cu} \) denote the current time period initialized as 1. \( T_{cu} \) controls \( C's \) responses to \( A's \) queries and the game terminates when \( T_{cu} = |T|. \) Intuitively, \( A \) can receive all secret keys \( sk_{ID} \), key updates \( ku_{ID,T} \), and decryption keys \( dk_{ID,T} \) if they are insufficient to derive \( dk_{ID',T} \), for target tuple \((ID', T').\)

The game proceeds as follows.

\( C \) runs \((MPK, sk_{kgc}) \leftarrow \text{Setup}(1^\lambda, L)\) and prepares \( SKList \), which initially contains \((kgc, sk_{kgc})\), and into which pairs of \((ID, sk_{ID})\) generated during the game are stored. When a new \( sk_{ID} \) is generated or existing ones are updated by executing \( \text{GenSK} \) or \( \text{KeyUp}, C \) stores \((ID, sk_{ID})\) or updates them in \( SKList \). Hereafter, we omit the descriptions of this addition/update for simplicity. Then, \( C \) executes \((ku_{kgc,1}, sk_{kgc}') \leftarrow \text{KeyUp}(MPK, T_{cu} = 1, sk_{kgc}, RL_{kgc,1} = \emptyset, \bot)\) to generate a key update for the initial time period \( T_{cu} = 1 \) and gives \((MPK, ku_{kgc,1})\) to \( A. \)

Then, \( A \) may adaptively make the following five types of a query to \( C. \)

Secret Key Generation Query: Upon a query \((ID, \ast) \in \mathcal{I}^{|ID|}\) from \( A, C \) checks if it holds that

- \((ID, \ast) \notin SKList\) and \((pa(ID), sk_{pa(ID)}) \in SKList\) for some \( sk_{pa(ID)}\).

This condition ensures that \( C \) has not still created \( sk_{ID} \) and \( C \) has already created \( sk_{pa(ID)}\). If the condition does not hold, \( C \) returns \( \bot \) to \( A. \) Otherwise, \( C \) executes \((sk_{ID}, sk_{pa(ID)}') \leftarrow \text{GenSK}(MPK, sk_{pa(ID)}, ID).\) If \(|ID| = 1, or 2 \leq |ID| \leq L - 1 and pa(ID) \notin RL_{pa(ID),T_{cu}}, \) then \( C \) executes \((ku_{ID,T}, sk_{ID}) \leftarrow \text{KeyUp}(PP, T, sk_{ID}, RL_{ID,T} := \emptyset, ku_{pa(ID),T})\) for \( T \in [T_{cu}]\) and returns \((ku_{ID,T})_{T \in [T_{cu}]} \) to \( A. \) If \( 2 \leq |ID| \leq L and pa(ID) \in RL_{pa(ID),T_{cu}}, \) then \( C \) executes \( RL_{pa(ID),T_{cu}} \leftarrow RL_{pa(ID),T_{cu}} \cup \{ID\} \) and returns nothing to \( A. \)

Note that all \( ID \) in the following queries (except the challenge query) must be “activated”, in the sense that \( sk_{ID} \) has already been generated via this query; thus, \((ID, sk_{ID}) \in SKList.\)

Secret Key Reveal Query: Until the challenge query, upon a query \((ID, \ast) \in \mathcal{I}^{|ID|}\) from \( A, C \) finds \( sk_{ID} \) from \( SKList \) and returns it to \( A. \) After the challenge query, \( C \) checks if it holds that

- If \( T_{cu} \geq T^* \) and \( ID \in \text{prefix}^+(ID^*), \) then \( ID' \in RL_{pa(ID'),T^*}, \) for some \( ID' \in \text{prefix}^+(ID).\)

This condition ensures that if \( ID \) is the ancestor of the challenge \( ID^*, \) \( ID \) or an ancestor of \( ID \) must be revoked by the challenge \( T^*. \) If the condition does not hold, \( C \) returns \( \bot \) to \( A;\)

\[\text{If } |ID'| = L, \text{ this step is skipped.}\]

\[\text{Here, } sk_{ID} \text{ is the latest secret key, i.e., the result of Step (2).}\]
otherwise, C finds sk_ID from SKList and returns it to A.

**Revoke & Key Update Query:** Until the challenge query, upon a query RL_{T_{cu}+1} \subseteq \mathcal{I}^{\leq L} (denoting the set of identities to be revoked in the next time period \(T_{cu} + 1\)) from A, C checks if the following conditions are satisfied simultaneously.

- If \(\mathcal{I} \subseteq \mathcal{I}^{\leq L-1}\) that appear in SKList.
- For all identities ID such that (ID, *) \in SKList and ID' \in \text{prefix}^+(ID), if ID' \in RL then ID \in RL.

The first condition ensures that once ID has been revoked, the same ID must be continuously revoked. The second condition ensures that ID must be revoked if one of its ancestor ID' \in \text{prefix}^+(ID) is revoked. After the challenge query, C also checks

- ID \in RL if ID \in \text{prefix}^+(ID^*), T_{cu} = T^* - 1, and sk_{ID'} for some ID' \in \text{prefix}^+(ID) has been revealed previously by the secret key reveal query.

The condition ensures that once A receives sk_ID for some ID \in \text{prefix}^+(ID^*), the same ID must be revoked at \(T^*\). If these conditions do not hold, then C returns \(\perp\) to A. Otherwise, C increments the current time period by \(T_{cu} \leftarrow T_{cu} + 1\) and executes the following operations (1) and (2) for all “activated” and non-revoked identities ID, i.e., ID \in \mathcal{I}^{\leq L-1} \cup \{kgc\}, (ID, *) \in SKList and ID \notin RL, in breadth-first order in the identity hierarchy.

1. Set RL_{ID,T_{cu}} \leftarrow RL \cap \mathcal{I}_{ID}, where we define \(\mathcal{I}_{kgc} := \mathcal{I}\).
2. Run \((ku_{ID,T_{cu},sk_{ID}}) \leftarrow \text{KeyUp}(MPK,T_{cu},sk_{ID},RL_{ID,T_{cu}},ku_{pa(ID),T_{cu}})\), where \(ku_{pa(kgc),T_{cu}} := \perp\).

Finally, C returns all of the generated \(\{ku_{ID,T_{cu}}\}_{(ID,*) \in \text{SKList}\setminus RL}\) to A.

**Decryption Key Reveal Query:** Until the challenge query, upon a query \((ID,T) \in \mathcal{I}^{\mid ID\mid} \times \mathcal{T}\) from A, C checks

- If \(T \leq T_{cu}\) holds.

After the challenge query, C also checks

- If \((ID,T) \neq (ID^*,T^*)\) holds.

If these conditions are not satisfied, then C returns \(\perp\) to A. Otherwise, C finds sk_ID from SKList, runs \(dk_{ID,T} \leftarrow \text{GenDK}(MPK,sk_{ID},ku_{pa(ID),T})\), and returns \(dk_{ID,T}\) to A.

**Challenge Query:** Note that A is permitted to make this query exactly once. Upon a query \((ID^*,T^*,M_0^*,M_1^*)\) such that \(|M_0^*| = |M_1^*|\) from A, C determines if the following conditions are satisfied simultaneously.

- If \(T^* \leq T_{cu}\), A has not submitted \((ID^*,T^*)\) as a decryption key reveal query.
- If \(T^* \leq T_{cu}\) and sk_ID for ID \in \text{prefix}^+(ID^*) has been revealed to A, then ID \in RL_{pa(ID),T^*-1}.

If these conditions are not satisfied, then C returns \(\perp\) to A. Otherwise, C selects a bit \(b \in \{0,1\}\) uniformly at random, runs \(ct^* \leftarrow \text{Enc}(MPK,ID^*,T^*,M_b^*)\), and returns the challenge ciphertext \(ct^*\) to A.

At some point, A outputs \(b' \in \{0,1\}\) as its guess for \(b\) and terminates.

This completes the description of the game. In this game, A’s adaptive security advantage is defined by \(\text{Adv}^{RHIBE}_{II,L,A}(\lambda) := 2 \cdot |\text{Pr}[b' = b] - 1/2|\).

**Definition 3.** We say that an RHIBE scheme \(\Pi\) of depth \(L\) satisfies adaptive security if the advantage \(\text{Adv}^{RHIBE}_{II,L,A}(\lambda)\) is negligible for all PPT adversaries A.
3 Proposed RHIBE Scheme

In this section, we propose an adaptively secure RHIBE scheme. First, we present the CS method in Section 3.1. Then, we present the proposed RHIBE scheme in Section 3.2. Finally, we prove the correctness of the scheme in Section 3.3.

3.1 CS Method

Before presenting the CS method, we summarize the notation of binary trees. Let $BT_{pa(ID)}$ denote a binary tree with $N$ leaves managed by a parent user $pa(ID)$. We use $\theta$ to denote a node in a binary tree. Especially, we use $\eta$ to denote a leaf node in a binary tree. For a leaf node $\eta$, let $Path(BT_{pa(ID)}, \eta)$ denote a path in a binary tree $BT_{pa(ID)}$ from the root node to the leaf node $\eta$.

In this paper, we describe the CS method as follows.

**CS.SetUp($1^\lambda, pa(ID)$) → $BT_{pa(ID)}$:** The setup algorithm takes the security parameter $1^\lambda$ and a parent identity $pa(ID) \in \mathcal{ID}$ as input, and outputs the description of a binary tree $BT_{pa(ID)}$ for $pa(ID)$.

**CS.Assign($BT_{pa(ID)}, \mathcal{AC}_{pa(ID)}, ID$) → ($\eta_{ID}, \mathcal{AC}'_{pa(ID)}$):** The assign algorithm takes binary tree $BT_{pa(ID)}$, a set of leaf nodes $\mathcal{AC}_{pa(ID)}$, and an identity $ID \in \mathcal{ID}$, and assigns $ID$ to a leaf node $\eta_{ID} \in \mathcal{L}_{pa(ID)} \setminus \mathcal{AC}_{pa(ID)}$ and updates $\mathcal{AC}'_{pa(ID)} \leftarrow \mathcal{AC}_{pa(ID)} \cup \{\eta_{ID}\}$. Finally, it outputs $\eta_{ID}$ and $\mathcal{AC}'_{pa(ID)}$.

**CS.Cover($BT_{pa(ID)}, \mathcal{RL}_{pa(ID), T}$) → $KUN_{pa(ID), T}$:** The cover algorithm takes a binary tree $BT_{pa(ID)}$ and a set of leaf nodes $\mathcal{RL}_{pa(ID), T}$, and outputs a set of nodes $KUN_{pa(ID), T}$.

**CS.Match($KUN_{pa(ID), T}, \eta_{ID}$) → $\theta$ or $\bot$:** The matching algorithm takes a set of nodes $KUN_{pa(ID), T}$ output by $CS.Cover$ and a leaf node $\eta_{ID}$ as input, and outputs $\theta \in KUN_{pa(ID), T} \cap Path(BT_{pa(ID)}, \eta_{ID})$ if such a node exists; otherwise, it outputs an invalid symbol $\bot$.

The CS method satisfies the following properties:

**Correctness:** For any leaf node $\eta_{ID} \in \mathcal{AC}_{pa(ID)} \setminus \mathcal{RL}_{pa(ID), T}$, it holds that $Path(BT_{pa(ID)}, \eta_{ID}) \cap KUN_{pa(ID), T} \neq \emptyset$.

**Security:** For any leaf node $\eta_{ID} \in \mathcal{RL}_{pa(ID), T}$, it holds that $Path(BT_{pa(ID)}, \eta_{ID}) \cap KUN_{pa(ID), T} = \emptyset$.

**Scalability:** It holds that $|KUN_{pa(ID), T}| = O(\|\mathcal{RL}_{pa(ID), T}\| \log(N/|\mathcal{RL}_{pa(ID), T}|))$.

**Remark 1.** In this paper, we did not define how $CS.Assign$ algorithm samples the leaf node $\eta_{ID}$ from $\mathcal{L}_{pa(ID)} \setminus \mathcal{AC}_{pa(ID)}$. In most $R(H)IBE$ schemes such as adaptively secure Emura et al.’s RHIBE schemes [ETW20], $\eta_{ID}$ should be sampled from $\mathcal{L}_{pa(ID)} \setminus \mathcal{AC}_{pa(ID)}$ uniformly at random so that their security proof works. In contrast, our security proof does not require any conditions for the distribution of $\eta_{ID}$. For example, we can set $\eta_{ID}$ as the leftmost leaf node in $\mathcal{L}_{pa(ID)} \setminus \mathcal{AC}_{pa(ID)}$.

3.2 Construction

Here, we provide an overview of our proposed RHIBE scheme. In our RHIBE scheme, each parent user $pa(ID)$ manages a binary tree $BT_{pa(ID)}$ → $CS.SetUp(1^\lambda, pa(ID))$ and assigns their children users ID to distinct leaf nodes $\eta_{ID}$ ← $CS.Assign(BT_{pa(ID)}, \mathcal{AC}_{pa(ID)}, ID)$. The secret key $sk_{ID}$ of the user $sk_{ID}$ contains the sub-secret keys $sk_{ID, \theta}$ associated with all nodes $\theta \in Path(BT_{pa(ID)}, \eta_{ID})$. The parent user $pa(ID)$ sets a set of leaf nodes $\mathcal{RL}_{pa(ID), T}$ so that $\eta_{ID} \in \mathcal{RL}_{pa(ID), T}$ hold iff children users ID are revoked at the time period $T$. The key update $ku_{pa(ID), T, \theta}$ associated with all nodes $\theta \in KUN_{pa(ID), T}$ of a parent user $pa(ID)$ contains sub-key updates $ku_{pa(ID), T, \theta}$ associated with all nodes $\theta \in KUN_{pa(ID), T}$. We designed the proposed RHIBE scheme so that children users ID can produce their decryption keys $dk_{ID, T}$ iff their sub-secret keys.
and their parent user $\text{pa}(\text{ID})$'s sub-key updates share the same node. Thus, the correctness of the CS method ensures that non-revoked users $\text{ID}$ can produce their decryption keys $\text{dk}_{\text{ID},T}$ properly, while the security of the CS method ensures that revoked users cannot produce them. Furthermore, the scalability of the CS method ensures that the size of the key update $\text{ku}_{\text{pa}(\text{ID}),T}$ grows logarithmically with the maximum number of children users $N$. For simplicity, we set $N = \lambda^{\Omega(1)}$ so that the parent user $\text{pa}(\text{ID})$ can register arbitrary polynomial numbers of children users in the RHIBE scheme.

We further provide an overview of how our RHIBE scheme achieves the revocation functionality by following the Lee and Park's RHIBE scheme [LP18]. In our RHIBE scheme, the KGC has the master secret key $\text{MSK} \in \mathbb{Z}_{p}^{k+1}$ as a part of $\text{sk}_{\text{kgc}}$. When the parent user $\text{pa}(\text{ID})$ creates a sub-secret key $\text{sk}_{\text{ID},\theta}$ or a sub-key update $\text{ku}_{\text{pa}(\text{ID}),\text{T},\theta}$ associated with a node $\theta \in \text{BT}_{\text{pa}(\text{ID})}$, the $\text{pa}(\text{ID})$ samples a delegation key $\text{delk}_{\text{pa}(\text{ID}),\theta} \leftarrow_R \mathbb{Z}_{p}^{k+1}$ associated with the node $\theta$. The sub-secret key $\text{sk}_{\text{ID},\theta}$ is an ID's HIBE secret key according to Chen and Gong's scheme achieved by setting $\text{delk}_{\text{pa}(\text{ID}),\theta}$ as a master secret key. The key update $\text{ku}_{\text{pa}(\text{ID}),\text{T}}$ consists of sub-key updates $\text{ku}_{\text{pa}(\text{ID}),\text{T},\theta}$ associated with all nodes $\theta \in \text{KU}_{\text{pa}(\text{ID}),\text{T}}$ and a helper key update $\text{ku}_{\text{ID},T}$ that is independent of any nodes $\theta \in \text{BT}_{\text{pa}(\text{ID})}$. To create a key update $\text{ku}_{\text{pa}(\text{ID}),\text{T}}$, the parent user $\text{pa}(\text{ID})$ samples an ephemeral delegation key $\text{delk}_{\text{pa}(\text{ID}),\theta} \leftarrow_R \mathbb{Z}_{p}^{k+1}$ and creates the sub-key update $\text{ku}_{\text{pa}(\text{ID}),\text{T},\theta}$ as a T's IBE secret key of Chen and Gong’s scheme by setting $-\text{delk}_{\text{pa}(\text{ID}),\theta}$ as a master secret key whereas the helper key update $\text{ku}_{\text{ID},T}$ as a multiplication of $\text{pa}(\text{ID})$'s HIBE secret key and T's IBE secret key of Chen and Gong’s scheme by setting $\text{MSK} + \text{delk}_{\text{pa}(\text{ID}),\theta}$ as a master secret key. That is, non-revoked users have the HIBE/IBE secret keys of Chen and Gong’s scheme with the master secret keys $\text{delk}_{\text{pa}(\text{ID}),\theta}$, $-\text{delk}_{\text{pa}(\text{ID}),\theta}$, and $\text{MSK} + \text{delk}_{\text{pa}(\text{ID}),\theta}$ for the same node $\theta$. Thus, by multiplying all the elements, the non-revoked users can produce decryption keys $\text{dk}_{\text{ID},T}$ with the master secret key $\text{MSK}$. In contrast, non-revoked users cannot cancel the delegation keys $\text{delk}_{\text{pa}(\text{ID}),\theta}$ from their own sub-secret keys and the parent user $\text{pa}(\text{ID})$'s key update. Thus, non-revoked users cannot produce their decryption keys.

Then, we propose the following RHIBE scheme.

**Setup$(1^{\lambda}) \rightarrow (\text{MPK}, \text{sk}_{\text{kgc}})$**: Run $(p, G_{1}, G_{2}, G_{T}, g_{1}, g_{2}, e) \leftarrow G(1^{\lambda})$ and sample $A \leftarrow \mathcal{D}_{k}$, uniformly random matrices $((V_{\ell})_{\ell \in [0, L+2]}, Z) \leftarrow_R (\mathbb{Z}_{p}^{(k+1)^{2}L+3} \times \mathbb{Z}_{p}^{k})$ and a random vector $k \leftarrow_R \mathbb{Z}_{p}^{k+1}$. Then, output

$$\text{MPK} := \left( [A]_{1}, (V_{\ell}^{T}A)_{1}, (Z)_{1}, (V_{\ell}Z)_{1}, (V_{\ell}Z)_{2} \right)_{\ell \in [0, L+2]}, [A^{T}k]_{T}$$

and $\text{sk}_{\text{kgc}} := (\text{MSK} := k, \text{BT}_{\text{kgc}})$, where $\text{MPK} \in G_{1}^{(k+1)^{2}L+3} \times G_{2}^{k}$. The KGC has the $\text{sk}_{\text{kgc}} := (\text{MSK} := k, \text{BT}_{\text{kgc}})$, where $\text{MPK} \in G_{1}^{(k+1)^{2}L+3} \times G_{2}^{k}$. The KGC has the $\text{sk}_{\text{kgc}} := (\text{MSK} := k, \text{BT}_{\text{kgc}})$, where $\text{MPK} \in G_{1}^{(k+1)^{2}L+3} \times G_{2}^{k}$.

**Enc$(\text{MPK}, \text{ID}, T, M) \rightarrow \text{ct}_{\text{ID}, T}$**: Sample $s \leftarrow_R \mathbb{Z}_{p}^{L}$, $(v_{0}, v_{1}, \ldots, v_{|\text{ID}|}, v_{L+1}) \leftarrow_R \mathbb{Z}_{p}^{|\text{ID}|+2}$, then output

$$\text{ct}_{\text{ID}, T} := (C_{0}, C_{1}, C_{1}', C_{2}, \text{tag}, \text{tag}') \in G_{1}^{k+1} \times (G_{1}^{k})^{L+3} \times G_{2}^{k} \times G_{2}^{(k+1)^{2}L+3} \times \mathcal{G}_{T}^{2}.$$ 

$$\text{tag} := v_{0} + v_{1}\text{id}_{1} + \cdots + v_{|\text{ID}|}\text{id}_{|\text{ID}|}, \quad \text{tag}' := v_{0} + v_{L+1}T, \quad C_{0} := [As]_{1},$$

$$C_{1} := [(V_{0} + \text{id}_{1}V_{1} \cdots + \text{id}_{|\text{ID}|}V_{|\text{ID}|} + \text{tag}V_{L+2})^{T}A]_{1},$$

$$C_{1}' := [(V_{0} + TV_{L+1} + \text{tag}'V_{L+2})^{T}A]_{1}, \quad C_{2} := M \cdot [s^{T}A^{T}k]_{T}.$$ 

**GenSK$(\text{MPK}, \text{sk}_{\text{pa}(\text{ID})}, \text{ID}) \rightarrow \text{sk}_{\text{ID}}$**: Run $(\eta_{\text{ID}}, \text{BT}_{\text{pa}(\text{ID})}^{'}) \leftarrow \text{CS.Assign}(\text{BT}_{\text{pa}(\text{ID})}, \text{ID})$. Parse

$$\text{sk}_{\text{kgc}} = (k, \text{BT}_{\text{kgc}}, (\theta, \text{delk}_{\text{kgc}, \theta})_{\theta \in A\mathcal{N}_{\text{kgc}}})$$

11
If there is a node $k \in \mathcal{N}_{\mathsf{pa}(ID)}$, sample a delegation key $\text{del}_k = k_{\mathsf{pa}(ID), \theta} \leftarrow R \mathbb{Z}_p^{k+1}$ and update $\mathsf{BT}_{\mathsf{pa}(ID)}$ by $\mathcal{N}_{\mathsf{pa}(ID)} \leftarrow \mathcal{N}_{\mathsf{pa}(ID)} \cup \{ \theta \}$ until $\mathsf{Path}(\mathsf{BT}_{\mathsf{pa}(ID)}, \eta_{\mathsf{ID}}) \subseteq \mathcal{N}_{\mathsf{pa}(ID)}$.

### Delegation Key Generation

**Sub-key Update Generation:** For each node $\theta \in \mathsf{Path}(\mathsf{BT}_{\mathsf{pa}(ID)}, \eta_{\mathsf{ID}})$, retrieve a delegation key $\text{del}_{\mathsf{pa}(ID), \theta} = k_{\mathsf{pa}(ID), \theta}$, sample $r_{\mathsf{ID}, \theta} \leftarrow R \mathbb{Z}_p^\ell$, and compute a sub-secret key $sk_{\mathsf{ID}, \theta} := (SK_{\mathsf{ID}, \theta, 0}, SK_{\mathsf{ID}, \theta, 1}, SK_{\mathsf{ID}, \theta, 2}, (SK_{\mathsf{ID}, \theta, 0})_{\ell \in [\mathsf{ID}+1, \mathsf{L}]) \in \mathbb{G}_2 \times (\mathbb{G}_2^{k+1})^{L-\mathsf{ID}+2}$.

$$
\begin{align*}
SK_{\mathsf{ID}, \theta, 0} &= [Z_{\mathsf{ID}, \theta}]_2, \\
SK_{\mathsf{ID}, \theta, 1} &= [k_{\mathsf{pa}(ID), \theta}]_2 \cdot \left( [V_0 + \mathbf{id}_1 V_1 + \cdots + \mathbf{id}_\mathsf{ID} V_{\mathsf{ID}}] Z_{\mathsf{ID}, \theta} \right)_2, \\
SK_{\mathsf{ID}, \theta, 2} &= [V_{L+2} Z_{\mathsf{ID}, \theta}]_2,
\end{align*}
$$

Finally, run $\mathsf{BT}_{\mathsf{ID}} \leftarrow \mathsf{CS.SetUp}(1^\lambda, \mathsf{ID})$, and output $\mathsf{ID}$’s secret key

$$sk_{\mathsf{ID}} := ((\theta, sk_{\mathsf{ID}, \theta})_{\theta \in \mathsf{Path}(\mathsf{BT}_{\mathsf{pa}(ID)}, \eta_{\mathsf{ID}})}, \mathsf{BT}_{\mathsf{ID}}),$$

and an updated secret key $sk'_{\mathsf{pa}(ID)}$

$$sk'_{\mathsf{pa}(ID)} = (k, B_{\mathsf{kgc}}', (\theta, \text{del}_{\mathsf{kgc}, \theta})_{\theta \in \mathcal{N}_{\mathsf{kgc}}})$$

if $\mathsf{pa}(\mathsf{ID}) = \mathsf{kgc}$, or

$$sk'_{\mathsf{pa}(ID)} = \left( (\theta, sk_{\mathsf{pa}(ID), \theta})_{\theta \in \mathsf{Path}(\mathsf{BT}_{\mathsf{pa}(ID)}, \eta_{\mathsf{pa}(ID)})}, \mathsf{BT}'_{\mathsf{pa}(ID)} \right) \left( (\theta, \text{del}_{\mathsf{pa}(ID), \theta})_{\theta \in \mathcal{N}'_{\mathsf{pa}(ID)}} \right)$$

otherwise.

### KeyUp

$\text{KeyUp}(\mathsf{MPK}, sk_{\mathsf{ID}, T}, \mathsf{RL}_{\mathsf{ID}, T}, ku_{\mathsf{pa}(ID), T}) \rightarrow (ku_{\mathsf{ID}, T}, sk'_{\mathsf{ID}})$: Run $\mathsf{KUN}_{\mathsf{ID}, T} \leftarrow \mathsf{CS.Cover}(\mathsf{BT}_{\mathsf{ID}}, \mathcal{R}_{\mathsf{ID}, T})$.

Parse

$$sk_{\mathsf{kgc}} = (k, B_{\mathsf{kgc}}', (\theta, \text{del}_{\mathsf{kgc}, \theta})_{\theta \in \mathcal{N}_{\mathsf{kgc}}})$$

or

$$sk_{\mathsf{ID}} = ((\theta, sk_{\mathsf{ID}, \theta})_{\theta \in \mathsf{Path}(\mathsf{BT}_{\mathsf{pa}(ID)}, \eta_{\mathsf{ID}})}, \mathsf{BT}_{\mathsf{ID}}, (\theta, \text{del}_{\mathsf{ID}, \theta})_{\theta \in \mathcal{N}_{\mathsf{ID}}})$$

if $\mathsf{ID} \neq \mathsf{kgc}$.

### Delegation Key Generation

If there is a node $\theta \in \mathsf{KUN}_{\mathsf{ID}, T} \setminus \mathcal{N}_{\mathsf{ID}}$, sample a delegation key $\text{del}_{\mathsf{ID}, \theta} := k_{\mathsf{ID}, \theta} \leftarrow R \mathbb{Z}_p^{k+1}$ and update $\mathsf{BT}_{\mathsf{ID}}$ by $\mathcal{N}_{\mathsf{ID}} \leftarrow \mathcal{N}_{\mathsf{ID}} \cup \{ \theta \}$ until $\mathsf{KUN}_{\mathsf{ID}, T} \subseteq \mathcal{N}_{\mathsf{ID}}$.

### Ephemeral Delegation Key Generation

If $\mathsf{ID} = \mathsf{kgc}$, skip this step. Otherwise, sample an ephemeral delegation key $\text{del}_{\mathsf{ID}, T} := k_{\mathsf{ID}, T} \leftarrow R \mathbb{Z}_p^{k+1}$.

### Sub-key Update Generation

For each $\theta \in \mathsf{KUN}_{\mathsf{ID}, T}$, retrieve a delegation key $\text{del}_{\mathsf{ID}, \theta} = k_{\mathsf{ID}, \theta}$ and ephemeral delegation key $\text{del}_{\mathsf{ID}, T} = k_{\mathsf{ID}, T}$, and proceed as follows:
Retrieve a master secret key $K_{pk} = k$, sample $t_{k_{gc}, T, \theta} \leftarrow_R \mathbb{Z}_p^k$ and compute a sub-key update $\kappa_{k_{gc}, T, \theta} := (K_{k_{gc}, T, \theta, 0}, K_{k_{gc}, T, \theta, 1}, K_{k_{gc}, T, \theta, 2}) \in \mathbb{G}_2^k \times (\mathbb{G}_2^{k+1})^2$:

$$K_{k_{gc}, T, \theta, 0} := [Zt_{k_{gc}, T, \theta}]_2,$$
$$K_{k_{gc}, T, \theta, 1} := [k - k_{k_{gc}, \theta}]_2 \cdot [(V_0 + TV_{L+1})Zt_{k_{gc}, T, \theta}]_2,$$
$$K_{k_{gc}, T, \theta, 2} := [V_{L+2}Zt_{k_{gc}, T, \theta}]_2.$$

Case of $ID \neq k_{gc}$: Sample $t_{ID, T, \theta} \leftarrow_R \mathbb{Z}_p^k$ and compute a sub-key update $\kappa_{ID, T, \theta} := (K_{ID, T, \theta, 0}, K_{ID, T, \theta, 1}, K_{ID, T, \theta, 2}) \in \mathbb{G}_2^k \times (\mathbb{G}_2^{k+1})^2$:

$$K_{ID, T, \theta, 0} := [Zt_{ID, T, \theta}]_2,$$
$$K_{ID, T, \theta, 1} := [k_{ID, \theta} + \bar{K}_{ID, T}]_2 \cdot [(V_0 + TV_{L+1})Zt_{ID, T, \theta}]_2,$$
$$K_{ID, T, \theta, 2} := [V_{L+2}Zt_{ID, T, \theta}]_2.$$  

**Helper Key Update Generation:** If $ID = k_{gc}$, skip this step. Otherwise, run $\text{GenDK}(MPK, sk_{ID}, \kappa_{pa(ID), T})$ algorithm to compute a helper decryption key $\bar{d}_{k_{gc}, T} = (DK_{ID, T, 0}, DK_{ID, T, 1}, DK_{ID, T, 2}, DK'_{ID, T, 2}, \bar{D}_{k_{gc}, T, \ell}, (\bar{D}_{k_{gc}, T, \ell})_{\ell \in \|ID+1, L\|})$ as in (2) or (3). Retrieve an ephemeral delegation key $\bar{d}_{\text{del}}_{ID, T} = \bar{K}_{ID, T}$, sample $\tilde{t}_{ID, T}, \tilde{t}'_{ID, T} \leftarrow_R \mathbb{Z}_p^k$ and compute a helper key update $\kappa_{ID, T} := (K_{ID, T, 0}, K_{ID, T, 1}, K_{ID, T, 2}, K_{ID, T, 2}, (\kappa_{ID, T, \ell})_{\ell \in \|ID+1, L\|}) \in \mathbb{G}_2^k \times (\mathbb{G}_2^{k+1})^{L-|ID|+4}$:

$$K_{ID, T, 0} := DK_{ID, T, 0} \cdot [\tilde{Z}_{ID, T}]_2 = [\tilde{Z}_{ID, T}]_2,$$
$$K_{ID, T, 0} := DK'_{ID, T, 0} \cdot [\tilde{Z}'_{ID, T}]_2 = [\tilde{Z}'_{ID, T}]_2,$$
$$K_{ID, T, 1} := [\bar{K}_{ID, T}]_2 \cdot DK_{ID, T, 1} \cdot [(V_0 + \text{id}_1 V_1 + \cdots + \text{id}_{|ID|} V_{|ID|})Z_{ID, T}]_2$$
$$\quad \cdot [(V_0 + TV_{L+1})Z_{ID, T}]_2$$
$$\quad = [k + \bar{K}_{ID, T}]_2 \cdot [(V_0 + \text{id}_1 V_1 + \cdots + \text{id}_{|ID|} V_{|ID|})Z_{ID, T}]_2$$
$$\quad \cdot [(V_0 + TV_{L+1})Z_{ID, T}]_2,$$
$$K_{ID, T, 2} := DK_{ID, T, 2} \cdot [V_{L+2}Z_{ID, T}]_2 = [V_{L+2}Z_{ID, T}]_2,$$
$$K_{ID, T, 2} := DK'_{ID, T, 2} \cdot [V_{L+2}Z'_{ID, T}]_2 = [V_{L+2}Z'_{ID, T}]_2,$$
$$\kappa_{ID, T, \ell} := [\bar{K}_{ID, T}]_2 \cdot [V_{L}Z_{ID, T}]_2 = [V_{L}Z_{ID, T}]_2,$$

where $\tilde{t}_{ID, T} = u_{ID, T} + \tilde{t}_{ID, T}$ and $\tilde{t}'_{ID, T} = u'_{ID, T} + \tilde{t}'_{ID, T}$.

Finally, output a key update and updated secret key

$$\kappa_{k_{gc}, T} = (\theta, \kappa_{k_{gc}, T, \theta})_{\theta \in \mathbb{K}_{\mathbb{M}}}, \quad \text{sk}'_{k_{gc}} = (k, B_{k_{gc}}', (\theta, \text{del}_{k_{gc}, \theta})_{\theta \in \mathbb{A}_{\mathbb{N}_{k_{gc}}}}).$$

if $ID = k_{gc}$, or

$$\kappa_{ID, T} = ((\theta, \kappa_{ID, T, \theta})_{\theta \in \mathbb{K}_{\mathbb{M}}}, \bar{K}_{ID, T}),$$
$$\text{sk}'_{ID} = ((\theta, \text{sk}_{ID, \theta})_{\theta \in \mathbb{Path}(B_{pa(ID), T})}, B_{k_{gc}}', (\theta, \text{del}_{ID, \theta})_{\theta \in \mathbb{A}_{\mathbb{N}_{T}}}).$$

otherwise.
GenDK(MPK, sk_ID, ku_{pa(ID),T}) \rightarrow dk_{ID,T} or \perp: Parse

sk_{ID} = ((\theta, sk_{ID,\theta})_{\theta \in \text{Path}(BT_{pa(ID),\eta_{ID}}, BT_{ID}, (\theta, delk_{ID,\theta})_{\theta \in AN_{ID}})}

and

\text{ku}_{kcg,\tau} = (\theta, \text{ku}_{kcg,\tau,\theta})_{\theta \in KU'_{kcg,\tau}}

if \text{pa(ID)} = kcg, or

\text{ku}_{pa(ID),\tau} = ((\theta, \text{ku}_{pa(ID),\tau,\theta})_{\theta \in KU'_{pa(ID),\tau}, \text{\overline{ku}_{pa(ID),\tau}}})

otherwise.

\textbf{Helper Decryption Key Generation:} Run CS.Match(KU'_{pa(ID),\tau,\eta_{ID}}) to find \tilde{\theta} \in KU'_{pa(ID),\tau} \cap \text{Path}(BT_{pa(ID),\eta_{ID}}) and proceed as follows:

\textbf{Case of} \text{pa(ID)} = kcg: Retrieve

sk_{ID,\tilde{\theta}} = (SK_{ID,\tilde{\theta},0}, SK_{ID,\tilde{\theta},1}, SK_{ID,\tilde{\theta},2}, (\tilde{SK}_{ID,\tilde{\theta},\ell})_{\ell \in [ID+1,L]})

kuk_{gc,\tilde{\theta}} = (KU_{pa(ID),\tilde{\theta},0}, KU_{pa(ID),\tilde{\theta},1}, KU_{pa(ID),\tilde{\theta},2})

sample \tilde{u}_{ID,\tau}, \tilde{u}'_{ID,\tau} \leftarrow R Z_{p}^{k}, and compute a helper decryption key \tilde{dk}_{ID,\tau} = (DK_{ID,\tau,0}, DK'_{ID,\tau,0}, DK_{ID,\tau,1}, DK_{ID,\tau,2}, DK'_{ID,\tau,2}, (\tilde{DK}_{ID,\tau,\ell})_{\ell \in [2,L]}) \in \mathbb{G}_2 \times (\mathbb{G}_{2}^{k+1})^{L+3}.

\begin{align*}
DK_{ID,\tau,0} & := SK_{ID,\tilde{\theta},0} \cdot [Z_{\tilde{u}_{ID,\tau}}]_2 = [Z_{u_{ID,\tau}}]_2, \\
DK'_{ID,\tau,0} & := KU_{kcg,\tau,\tilde{\theta},0} \cdot [Z'_{\tilde{u}_{ID,\tau}}]_2 = [Z'_{u_{ID,\tau}}]_2, \\
DK_{ID,\tau,1} & := SK_{ID,\tilde{\theta},1} \cdot KU_{kcg,\tau,\tilde{\theta},1} \cdot [(V_0 + id_1V_1)Z_{\tilde{u}_{ID,\tau}}]_2 \\
& \quad \cdot [(V_0 + TV_{L+1})Z'_{\tilde{u}_{ID,\tau}}]_2 \\
& = [k]_2 \cdot [(V_0 + id_1V_1)Z_{u_{ID,\tau}}]_2 \\
& \quad \cdot [(V_0 + TV_{L+1})Z'_{u_{ID,\tau}}]_2,
DK_{ID,\tau,2} & := SK_{ID,\tilde{\theta},2} \cdot [V_{L+2}Z_{\tilde{u}_{ID,\tau}}]_2 = [V_{L+2}Z_{u_{ID,\tau}}]_2, \\
DK'_{ID,\tau,2} & := KU_{kcg,\tau,\tilde{\theta},2} \cdot [V_{L+2}Z'_{\tilde{u}_{ID,\tau}}]_2 = [V_{L+2}Z'_{u_{ID,\tau}}]_2, \\
\tilde{DK}_{ID,\tau,\ell} & := \tilde{SK}_{ID,\tilde{\theta},\ell} \cdot [V_{\ell}Z_{\tilde{u}_{ID,\tau}}]_2 = [V_{\ell}Z_{u_{ID,\tau}}]_2,
\end{align*}

where u_{ID,\tau} = r_{ID,\tilde{\theta}} + \tilde{u}_{ID,\tau} and u'_{ID,\tau} = t_{kcg,\tau,\tilde{\theta}} + \tilde{u}'_{ID,\tau}.

\textbf{Case of} \text{pa(ID)} \neq kcg: Retrieve

sk_{ID,\tilde{\theta}} = (SK_{ID,\tilde{\theta},0}, SK_{ID,\tilde{\theta},1}, SK_{ID,\tilde{\theta},2}, (\tilde{SK}_{ID,\tilde{\theta},\ell})_{\ell \in [ID+1,L]})

kuk_{pa(ID),\tilde{\theta}} = (KU_{pa(ID),\tilde{\theta},0}, KU_{pa(ID),\tilde{\theta},1}, KU_{pa(ID),\tilde{\theta},2})

\text{\overline{ku}_{pa(ID),\tau}} = \left(\begin{array}{c}
\overline{KU}_{pa(ID),\tau,0} \\
\overline{KU}_{pa(ID),\tau,0} \\
\overline{KU}_{pa(ID),\tau,1} \\
\overline{KU}_{pa(ID),\tau,2} \\
\overline{KU}_{pa(ID),\tau,2} \end{array}\right)_{\ell \in [pa(ID)+1,L]}.
sample \( \tilde{u}_{\mathit{ID},T}, \tilde{u}'_{\mathit{ID},T} \leftarrow R Z_{\mathbb{F}}^k \), and compute a helper decryption key \( \tilde{K}_{\mathit{ID},T} := (DK_{\mathit{ID},T,0}, DK'_{\mathit{ID},T,0}, DK_{\mathit{ID},T,1}, DK'_{\mathit{ID},T,1}, DK_{\mathit{ID},T,2}, DK'_{\mathit{ID},T,2}, (\tilde{K}_{\mathit{ID},T,\ell})_{\ell \in \lbrack \lbrack |\mathit{ID}| + 1, L \rbrack \rbrack} ) \in \mathbb{G}_2^k \times (\mathbb{G}_2^{k+1})^{L-|\mathit{ID}|+4} \):

\[
\begin{align*}
DK_{\mathit{ID},T,0} &:= SK_{\mathit{ID},\bar{\theta},0} \cdot KU_{\mathit{pa}(\mathit{ID}),T,0} \cdot [Z\tilde{u}_{\mathit{ID},T}]_2 = [Zu_{\mathit{ID},T}]_2, \\
DK'_{\mathit{ID},T,0} &:= KU_{\mathit{pa}(\mathit{ID}),T,\bar{\theta},0} \cdot \tilde{K}U_{\mathit{pa}(\mathit{ID}),T,0} \cdot [Z\tilde{u}_{\mathit{ID},T}]_2 = [Zu_{\mathit{ID},T}]_2, \\
DK_{\mathit{ID},T,1} &:= SK_{\mathit{ID},\bar{\theta},1} \cdot KU_{\mathit{pa}(\mathit{ID}),T,\bar{\theta},1} \cdot \tilde{K}U_{\mathit{pa}(\mathit{ID}),T,1} \cdot [V_{0 + \mathit{ID}V_{1} + \cdots + \mathit{ID}V_{|\mathit{ID}|}]}Z\tilde{u}_{\mathit{ID},T}]_2 \\
&\quad \cdot [(V_0 + TV_{L+1})Z\tilde{u}_{\mathit{ID},T}]_2 \\
&= [k_2] \cdot [(V_0 + id_{1}V_{1} + \cdots + id_{|\mathit{ID}|}V_{|\mathit{ID}|})Zu_{\mathit{ID},T}]_2 \\
&\quad \cdot [(V_0 + TV_{L+1})Zu_{\mathit{ID},T}]_2,
DK'_{\mathit{ID},T,1} &:= KU_{\mathit{pa}(\mathit{ID}),T,\bar{\theta},2} \cdot \tilde{K}U_{\mathit{pa}(\mathit{ID}),T,2} \cdot [V_{L+2Zu_{\mathit{ID},T}]_2, \\
DK_{\mathit{ID},T,2} &:= SK_{\mathit{ID},\bar{\theta},2} \cdot \tilde{K}K_{\mathit{pa}(\mathit{ID}),T,2} \cdot [V_{L+2Zu_{\mathit{ID},T}]_2, \\
DK'_{\mathit{ID},T,2} &:= KU_{\mathit{pa}(\mathit{ID}),T,\bar{\theta},2} \cdot \tilde{K}U_{\mathit{pa}(\mathit{ID}),T,2} \cdot [V_{L+2Z\tilde{u}_{\mathit{ID},T}]_2, \\
\tilde{D}K_{\mathit{ID},T,\ell} &:= \tilde{S}K_{\mathit{ID},\bar{\theta},\ell} \cdot \tilde{K}U_{\mathit{pa}(\mathit{ID}),T,\ell} \cdot [V_{\ell Z\tilde{u}_{\mathit{ID},T}]_2 = [V_{\ell Zu_{\mathit{ID},T}]_2, 

\end{align*}
\]

where \( u_{\mathit{ID},T} = r_{\mathit{ID},\bar{\theta}} + \tilde{t}_{\mathit{pa}(\mathit{ID}),T} + \tilde{u}_{\mathit{ID},T} \) and \( u'_{\mathit{ID},T} = t_{\mathit{pa}(\mathit{ID}),T,\bar{\theta}} + \tilde{t}_{\mathit{pa}(\mathit{ID}),T} + \tilde{u}'_{\mathit{ID},T} \).

Finally, output \( dk_{\mathit{ID},T} := (DK_{\mathit{ID},T,0}, DK'_{\mathit{ID},T,0}, DK_{\mathit{ID},T,1}, DK'_{\mathit{ID},T,1}, DK'_{\mathit{ID},T,2}, DK'_{\mathit{ID},T,2} ) \in (G_2^k)^2 \times (G_2^{k+1})^3 \).

Dec(MPK, c_{\mathit{ID},T}, dk_{\mathit{ID},T}) \rightarrow M: Parse \ c_{\mathit{ID},T} := (C_0, C_1, C'_1, C_2, \mathit{tag}, \mathit{tag}' ) and \ dk_{\mathit{ID},T} := (DK_{\mathit{ID},T,0}, DK_{\mathit{ID},T,0}, DK_{\mathit{ID},T,1}, DK_{\mathit{ID},T,2}, DK'_{\mathit{ID},T,2} ) \rightarrow M = C_2 \cdot \frac{e(C_1, DK_{\mathit{ID},T,0}) \cdot e(C'_1, DK'_{\mathit{ID},T,0})}{e(C_0, DK_{\mathit{ID},T,1} \cdot DK'_{\mathit{ID},T,2} \cdot (DK'_{\mathit{ID},T,2})^{\mathit{tag}' })}.

### 3.3 Correctness

The correctness of the CS method ensures that \( \text{CS.Match}(KU_{\mathit{pa}(\mathit{ID}),T}, \eta_{\mathit{ID}}) \) does not output \( \bot \), and there is a node \( \bar{\theta} \in KU_{\mathit{pa}(\mathit{ID}),T} \cap \text{Path}(\mathit{BT}_{\mathit{pa}(\mathit{ID}),T}) \) for the non-revoked user \( \mathit{ID} \). Since all \( sk_{\mathit{ID},\bar{\theta}} \) and \( ku_{\mathit{ID},T,\bar{\theta}} \) are computed directly, it is clear that they follow the distributions as we specified above. In contrast, we have to check that all \( \tilde{K}_{\mathit{ID},T} \) and \( dk_{\mathit{ID},T} \) created by using \( sk_{\mathit{ID},\bar{\theta}} \) and \( ku_{\mathit{ID},T,\bar{\theta}} \) or \( ku_{\mathit{pa}(\mathit{ID}),T,\bar{\theta}} \tilde{K}U_{\mathit{pa}(\mathit{ID}),T} \) follow the aforementioned distributions. Therefore, we first check that the helper decryption key \( \tilde{K}_{\mathit{ID},T} = (DK_{\mathit{ID},T,0}, DK'_{\mathit{ID},T,0}, DK_{\mathit{ID},T,1}, DK_{\mathit{ID},T,2}, DK_{\mathit{ID},T,2}, (\tilde{K}_{\mathit{ID},T,\ell})_{\ell \in \lbrack \lbrack 2, L \rbrack \rbrack} ) \) follows the distribution as specified in (2) and (3). In the following part of this section, we check the distribution of \( DK_{\mathit{ID},T,1} \); the validity of the other elements \( (DK_{\mathit{ID},T,0}, DK'_{\mathit{ID},T,0}, DK'_{\mathit{ID},T,2}, DK'_{\mathit{ID},T,2}) \) can be checked similarly.

**Case of \( pa(\mathit{ID}) = kgc \):** Since \( sk_{\mathit{ID},\bar{\theta}} := (SK_{\mathit{ID},\bar{\theta},0}, SK_{\mathit{ID},\bar{\theta},1}, SK_{\mathit{ID},\bar{\theta},2}) \cdot (\tilde{S}K_{\mathit{ID},\bar{\theta},\ell})_{\ell \in \lbrack \lbrack |\mathit{ID}| + 1, L \rbrack \rbrack} ) \) and \( ku_{\mathit{kgc},T,\bar{\theta}} := (KU_{\mathit{kgc},T,\bar{\theta},0}, KU_{\mathit{kgc},T,\bar{\theta},1}, KU_{\mathit{kgc},T,\bar{\theta},2}) \) follow the aforementioned distributions, we have

\[
\begin{align*}
DK_{\mathit{ID},T,1} &:= SK_{\mathit{ID},\bar{\theta},1} \cdot KU_{\mathit{kgc},T,\bar{\theta},1} \cdot [(V_0 + id_{1}V_{1})Z\tilde{u}_{\mathit{ID},T}]_2 \cdot [(V_0 + TV_{L+1})Z\tilde{u}_{\mathit{ID},T}]_2 \\
&= [k_{\mathit{kgc},\bar{\theta}}] \cdot [(V_0 + id_{1}V_{1})Z\tilde{u}_{\mathit{ID},T}]_2 \cdot [k - k_{\mathit{kgc},\bar{\theta}}] \cdot [(V_0 + TV_{L+1})Z\tilde{u}_{\mathit{ID},T}]_2
\end{align*}
\]
\[DK_{ID,T,1} = SK_{ID,\tilde{\theta},1} \cdot \overline{KU}_{pa(ID),T,\tilde{\theta},1} \cdot \overline{KU}_{pa(ID),T,\mid ID} \cdot (V_0 + \text{id}_{ID} V_1 + \cdots + \text{id}_{ID} V_{|ID|}) Z \tilde{u}_{ID,T} \cdot [(V_0 + TV_{L+1}) Z \tilde{u}_{ID,T}]_2\]

as we specified in (3).

Thus, \(dK_{ID,T} = (DK_{ID,T,0}, DK'_{ID,T,0}, DK_{ID,T,1}, DK_{ID,T,2}, DK'_{ID,T,2})\) follows the distribution as specified above.

Next, we check that the helper key update \(\overline{KU}_{ID,T} = (\overline{KU}_{ID,T,0}, \overline{KU}_{ID,T,0}, \overline{KU}_{ID,T,1}, \overline{KU}_{ID,T,2}, \overline{KU'}_{ID,T,1}, \overline{KU'}_{ID,T,2}, \overline{KU'}_{ID,T,0}, \overline{KU'}_{ID,T,1}, \overline{KU'}_{ID,T,2}, \overline{KU'}_{ID,T,3}, \overline{KU'}_{ID,T,4}, \overline{KU'}_{ID,T,5}, \overline{KU'}_{ID,T,6})\) follows the distribution as specified in (1). In the following, we check the distribution of \(\overline{KU}_{ID,T,1}\), whereas the validity of the other elements \(\overline{KU}_{ID,T} = (\overline{KU}_{ID,T,0}, \overline{KU}_{ID,T,0}, \overline{KU}_{ID,T,1}, \overline{KU}_{ID,T,2}, \overline{KU}_{ID,T,3}, \overline{KU}_{ID,T,4}, \overline{KU}_{ID,T,5}, \overline{KU}_{ID,T,6})\) can be checked in the same manner. When we assume that \(dK_{ID,T} = (DK_{ID,T,0}, DK'_{ID,T,0}, DK_{ID,T,1}, DK_{ID,T,2}, DK'_{ID,T,2}, (DK_{ID,T,3}, DK'_{ID,T,3})_{\ell \in [2, L]})\) follows the distribution as specified in (2) and (3), we have

\[\overline{KU}_{ID,T,1} = [K_{ID,T}]_2 \cdot DK_{ID,T,1} \cdot [(V_0 + \text{id}_{ID} V_1 + \cdots + \text{id}_{ID} V_{|ID|}) Z \tilde{u}_{ID,T}]_2 \cdot [(V_0 + TV_{L+1}) Z \tilde{u}'_{ID,T}]_2\]

as we specified in (3).
as we specified in (1).

Finally, we check that the decryption succeeds. Since we have

\[
e(C_1, DK_{ID,T,0}) \\
e = [(\langle V_0 + 1d_1 V_1 + \cdots + 1d_{ID} V_{ID} \rangle)^T A]_1 \cdot [V_{L+2, As}]_{1,tag} \cdot [Zu_{ID,T}]_2 \\
e = [(\langle A \rangle^T (V_0 + 1d_1 V_1 + \cdots + 1d_{ID} V_{ID}) Zu_{ID,T}]_T \cdot [(\langle A \rangle^T V_{L+2} Zu_{ID,T}]_T, \\
e e(C', DK'_{ID,T,0}) \\
e = e([\langle V_0 + TV_{L+1} \rangle A]_1 \cdot [V_{L+2, As}]_{1,tag} \cdot [Zu_{ID,T}]_2) \\
e = [(\langle A \rangle^T (V_0 + TV_{L+1}) Zu_{ID,T}]_T \cdot [(\langle A \rangle^T V_{L+2} Zu_{ID,T}]_T, \\
e = e(C_0, DK_{ID,T,1}) \\
e = e([\langle A \rangle^T k]_T \cdot [(\langle A \rangle^T (V_0 + 1d_1 V_1 + \cdots + 1d_{ID} V_{ID}) Zu_{ID,T}]_T \cdot [(\langle A \rangle^T (V_0 + TV_{L+1}) Zu_{ID,T}]_T, \\
e = e(C_0, DK_{ID,T,2} \cdot (DK'_{ID,T,2})_{tag}^T) \\
e = e([\langle A \rangle^T \cdot (V_{L+2} Zu_{ID,T}]_T \cdot [(\langle A \rangle^T V_{L+2} Zu_{ID,T}]_T,)
\]

it holds that

\[
C_2 \cdot \frac{e(C_1, DK_{ID,T,0}) \cdot e(C', DK'_{ID,T,0})}{e(C_0, DK_{ID,T,1} \cdot DK_{ID,T,2} \cdot (DK'_{ID,T,2})_{tag}^T)} = M.
\]

4 Main Theorem

The proposed RHIBE scheme in Section 3.2 achieves the adaptive security according to the following theorem.

**Theorem 1.** The proposed RHIBE scheme satisfies adaptive security if the MDDH assumption holds in \(G_1 \) and \(G_2\). Specifically, for any PPT adversary \(A\) making at most \(Q_{gen}\) secret key generation queries, there exists a reduction algorithm \(B_0 \) and \(B_{1,j}, B_{2,j}\) for \(j \in [6]\) such that

\[
Adv_{\text{RHIBE}}^{MDDH-G_1}(\lambda) \leq Adv_{B_0}^{MDDH-G_1}(\lambda) + Q_{gen} \sum_{i \in \{0, 1\}} \sum_{j \in [2]} Adv_{B_{1,i+j}}^{MDDH-G_2}(\lambda) + \sum_{j \in [4]} Adv_{B_{2,j}}^{MDDH-G_2}(\lambda) \\
+ |T| \cdot \sum_{j \in [2]} \left(Adv_{B_{1,2+j}}^{MDDH-G_2}(\lambda) + Adv_{B_{2,4+j}}^{MDDH-G_2}(\lambda)\right) + O\left(Q_{gen} |T| / p\right)
\]

and \(T(B_0) \approx \max_{j \in [6]} \{T(B_{1,j}), T(B_{2,j})\} \approx T(A) + k^2 Q_{gen} |T| \cdot \text{poly}(\lambda, L)\), where \(\text{poly}(\lambda, L)\) is independent of \(T(A)\).
4.1 Auxiliary Distributions

To prove theorem 1, we introduce the following semi-functional distributions of the challenge ciphertext $ct^*$, KGC’s sub-key updates $ku_{kgc,T,\theta}$, ID’s helper key updates $ku_{ID,T}$ such that $|ID| \geq 1$, and decryption keys $dk_{ID,T}$.

**Semi-functional Ciphertext:** A semi-functional ciphertext for the target for $(ID^*, T^*)$ and a plaintext $M_{\text{coin}}^*$ is defined as $ct^* = (C_0, C_1, C_2, \text{tag}, \text{tag}')$:

\[
\begin{align*}
tag & := v_0 + v_1id_1^* + \cdots + v_{|ID^*|}id_1^{|ID^*|}, & \quad \text{tag}' & := v_0 + v_{L+1}T^*, \\
C_0 & := [c]_1, & \\
C_1 & := [(V_0 + id_1^*V_1 + \cdots + id_1^{|ID^*|}V_{|ID^*|} + \tag V_{L+2})^T c]_1, \\
C_1' & := [(V_0 + T^*V_{L+1} + \tag' V_{L+2})^T c]_1, \\
C_2 & := M_{\text{coin}} \cdot [c]^T MSK]_T, \\
\end{align*}
\]

where $(v_0, v_1, \ldots, v_{|ID|}, v_{L+1}) \leftarrow_R \mathbb{Z}_p^{|ID|+2}$ and $c \leftarrow_R \mathbb{Z}_p k+1$. Here, the boxed parts denote the change from the normal ciphertext.

**Semi-functional KGC’s Key Updates:** A semi-functional KGC’s key update $ku_{kgc,T}$ for $T$ is defined with the following sub-key updates $ku_{kgc,\theta} = (K_U_{kgc,T,\theta,0}, K_U_{kgc,T,\theta,1}, K_U_{kgc,T,\theta,2})$:

\[
\begin{align*}
K_U_{kgc,T,\theta,0} & := [Zt_{kgc,T,\theta}]_2, \\
K_U_{kgc,T,\theta,1} & := [MSK + \frac{\alpha a^\perp}{a^\perp} - \text{delk}_{kgc,\theta}]_2 \cdot [(V_0 + TV_{L+1})Zt_{kgc,T,\theta}]_2, \\
K_U_{kgc,T,\theta,2} & := [V_{L+2}Zt_{kgc,T,\theta}]_2,
\end{align*}
\]

where $t_{kgc,T,\theta} \leftarrow_R \mathbb{Z}_p k$ and $\alpha \leftarrow_R \mathbb{Z}_p k$ is shared by all semi-functional $ku_{kgc,T}$, $ku_{ID,T}$, and $dk_{ID,T}$ unless stated otherwise. Here, the boxed part denotes the change from the normal KGC’s key update.

**Semi-functional Helper Key Updates:** A semi-functional helper key update $ku_{ID,T}$ for $(ID, T)$ is defined as $ku_{ID,T} = (K_U_{ID,T,0}, K_U'_{ID,T,0}, K_U'_{ID,T,1}, K_U'_{ID,T,2}, K_U'_{ID,T,2}, K_U'_{ID,T,\ell})_{\ell \in [ID+1, L]}$:

\[
\begin{align*}
K_U_{ID,T,0} & := [Zt_{ID,T}]_2, & K_U'_{ID,T,0} & := [Zt'_{ID,T}]_2, \\
K_U_{ID,T,1} & := [MSK + \frac{\alpha a^\perp}{a^\perp} + \text{delk}_{ID,T}]_2 \\
& \cdot [(V_0 + id_1^*V_1 + \cdots + id_{|ID|}^*V_{|ID|})Zt_{ID,T}]_2, \\
& \cdot [(V_0 + TV_{L+1})Zt'_{ID,T}]_2, \\
K_U_{ID,T,2} & := [V_{L+2}Zt_{ID,T}]_2, & K_U'_{ID,T,2} & := [V_{L+2}Zt'_{ID,T}]_2, \\
\bar{K}_U_{ID,T,\ell} & := [V_\ell Zt_{ID,T}]_2,
\end{align*}
\]

where $t_{ID,T,\theta}, t'_{ID,T,\theta} \leftarrow_R \mathbb{Z}_p k$, $\bar{K}_{ID,T} \leftarrow_R \mathbb{Z}_p k+1$, and $\alpha \leftarrow_R \mathbb{Z}_p k$ is shared by all semi-functional $ku_{kgc,T}$, $ku_{ID,T}$, and $dk_{ID,T}$ unless stated otherwise. Here, the boxed part denotes the change from the normal helper key update.

**Semi-functional Decryption Keys:** A semi-functional decryption key for $(ID, T)$ is defined as
\[ dk_{\text{ID},T} = (DK_{\text{ID},T,0}, DK'_{\text{ID},T,0}, DK_{\text{ID},T,1}, DK_{\text{ID},T,2}, DK'_{\text{ID},T,2}) : \]
\[
DK_{\text{ID},T,0} := [Zu_{\text{ID},T}]_2, \quad DK'_{\text{ID},T,0} := [Zu'_{\text{ID},T}]_2, \\
DK_{\text{ID},T,1} := |\text{MSK} + [\alpha a^+]_2 \cdot ([V_0 + id_1 V_1 + \cdots + id_{|\text{ID}|} V_{|\text{ID}|}] Zu_{\text{ID},T}]_2 \\
\cdot ([V_0 + TV_{L+1}] Zu'_{\text{ID},T}]_2, \\
DK_{\text{ID},T,2} := [V_{L+2} Zu_{\text{ID},T}]_2, \quad DK'_{\text{ID},T,2} := [V_{L+2} Zu'_{\text{ID},T}]_2, \\
\bar{D}K_{\text{ID},T,\ell} := [V_\ell Zu_{\text{ID},T}]_2, \tag{7}
\]

where \( u_{\text{ID},T}, u'_{\text{ID},T} \leftarrow_R Z_p^k \) and \( \alpha \leftarrow_R Z_p^* \) is shared by all semi-functional \( k_{\text{KG},T}, \bar{k}_{\text{ID},T}, \) and \( dk_{\text{ID},T} \) unless stated otherwise. Here, the boxed part denotes the change from the normal decryption key.

In brief, the above semi-functional ciphertext \( ct^* \) is the same as the normal ciphertext when we set \( c = A s \), where \( s \leftarrow_R Z_p^k \), while the above semi-functional \( k_{\text{KG},T,\theta}, \bar{k}_{\text{ID},T}, \) and \( dk_{\text{ID},T} \) are the same as the normal ones when we set \( \alpha = 0 \). If KGC’s sub-key updates \( k_{\text{KG},T,\theta} \) or ID’s helper key updates \( \bar{k}_{\text{ID},T} \) are semi-functional, the decryption keys \( dk_{\text{ID},T} \) computed by them with the normal sub-secret keys \( sk_{\text{ID},\theta} \) become semi-functional. Both normal and semi-functional decryption keys \( dk_{\text{ID},T} \) can correctly decrypt normal ciphertexts, whereas the semi-functional decryption keys \( dk_{\text{ID},T} \) cannot correctly decrypt the semi-functional ciphertexts. By following the standard dual system argument [CGW15, CG17, CW14, Wat09], we first change the challenge ciphertext \( ct^* \) to be semi-functional; then, we change a part of keys that \( A \) receives to be semi-functional. To this end, the semi-functional distributions of secret keys \( sk_{\text{ID}}, k \) updates \( k_{\text{ID},T} \), and decryption keys \( dk_{\text{ID},T} \) are defined so that the MSK is masked by \( \alpha a^+ \). In other words, we do not define semi-functional distributions for sub-secret keys \( sk_{\text{ID},\theta} \) and ID’s sub-key updates \( k_{\text{ID},T,\theta} \) such that \( |\text{ID}| \geq 1 \) since they do not contain MSK. If all information of the MSK that \( A \) receives is masked by \( \alpha a^+ \), the standard dual system argument [CGW15, CG17, CW14] enables us to show that the plaintext \( M_{\text{coin}}^* \) is information theoretically hidden. The main technical hurdle to proving the security is to change all of KGC’s sub-key updates \( k_{\text{KG},T,\theta} \), ID’s helper key updates \( \bar{k}_{\text{ID},T} \), and decryption keys \( dk_{\text{ID},T} \) that \( A \) receive to be semi-functional. Care should be taken that \( A \) can receive \( k_{\text{KG},T,\theta} \) for \( T = T^*, \bar{k}_{\text{ID},T} \) for \( \text{ID} \in \text{prefix}^+(\text{ID}^*) \land T = T^*, \) and \( dk_{\text{ID},T} \) for \( \text{ID} \in \text{prefix}^+(\text{ID}^*) \setminus \{\text{ID}^*\} \land T = T^* \), which the standard dual system argument cannot change to be semi-functional. We will use the semi-functional randomness switching for changing them to semi-functional.

4.2 Proof of Main Theorem

We conclude this section by introducing the way we prove Theorem 1. By following previous security proofs of RHIBE (e.g., [ETW20, LP18, SE15]), we divide \( A \)’s attack strategy into the following two types.

Type-I Adversary: \( A \) is called Type-I if it makes secret key reveal queries on some \( \text{ID} \in \text{prefix}^+(\text{ID}^*) \).

Type-II Adversary: \( A \) is called Type-II if it does not make secret key reveal queries on any \( \text{ID} \in \text{prefix}^+(\text{ID}^*) \).

Remark 2. To be precise, previous security proofs of RHIBE (e.g., [ETW20, LP18, SE15]) further divides the Type-I adversary into \( L \) types depending on the value \( \ell^* \in [L] \) so that \( A \) receive \( sk_{\text{ID}^*_{\ell^*}} \), whereas \( A \) does not receive \( sk_{\text{ID}^*_{\ell}} \) for any \( \ell \in [\ell^* - 1] \). Since our proof does not require the division, our proof saves the reduction loss by a factor \( O(L) \).
Note that the Type-I adversary and Type-II adversary are mutually exclusive and cover all the possible strategies of \( A \). We prove the adaptive security of the proposed RHIBE scheme against the Type-I adversary and Type-II adversary in distinct ways and obtain the following results.

**Lemma 1** (Adaptive Security against the Type-I Adversary). The proposed RHIBE scheme satisfies adaptive security against the Type-I adversary if the MDDH assumption holds in \( G_1 \) and \( G_2 \). Specifically, for any PPT Type-I adversary \( A \) making at most \( Q_{\text{gen}} \) secret key generation queries, there exists reduction algorithms \( B_0 \) and \( B_{II,j} \) for \( j \in \{6\} \) such that

\[
\text{Adv}_{RHIBE}(\lambda) \leq \text{Adv}_{B_0}^{MDDH-G_1}(\lambda) + Q_{\text{gen}} \left( \sum_{i \in \{0,4\}} \sum_{j \in \{2\}} \text{Adv}_{B_{II,i+j}}^{MDDH-G_2}(\lambda) + \left| T \right| \sum_{j \in \{2\}} \text{Adv}_{B_{II,i+j}}^{MDDH-G_2}(\lambda) + \frac{1}{p} \right)
\]

and \( T(B_0) \approx \max_{j \in \{6\}} T(B_{II,j}) \approx T(A) + k^2 Q_{\text{gen}} |T| \cdot \text{poly}(\lambda, L) \), where \( \text{poly}(\lambda, L) \) is independent of \( T(A) \).

**Lemma 2** (Adaptive Security against the Type-II Adversary). The proposed RHIBE scheme satisfies adaptive security against the Type-II adversary if the MDDH assumption holds in \( G_1 \) and \( G_2 \). Specifically, for any PPT Type-II adversary \( A \) making at most \( Q_{\text{gen}} \) secret key generation queries, there exists a reduction algorithm \( B_0 \) and \( B_{II,j} \) for \( j \in \{6\} \) such that

\[
\text{Adv}_{RHIBE}(\lambda) \leq \text{Adv}_{B_{II,i+j}}^{MDDH-G_1}(\lambda) + Q_{\text{gen}} \left( \sum_{j \in \{4\}} \text{Adv}_{B_{II,j}}^{MDDH-G_2}(\lambda) + |T| \sum_{j \in \{2\}} \text{Adv}_{B_{II,i+j}}^{MDDH-G_2}(\lambda) \right) + O \left( \frac{Q_{\text{gen}} |T|}{p} \right).
\]

and \( T(B_0) \approx \max_{j \in \{6\}} T(B_{II,j}) \approx T(A) + Q_{\text{gen}} |T| \cdot \text{poly}(\lambda, L) \), where \( \text{poly}(\lambda, L) \) is independent of \( T(A) \).

We omit the proof of Theorem 1 since it is clear from Lemmata 1 and 2. Since the Type-I adversary can receive \( \text{sk}_{ID} \) for some \( ID \in \text{prefix}^+(ID^*) \) as oppose to the Type-II adversary, the proof against the Type-I adversary is more complicated than the proof against the Type-II adversary; thus, we first prove Lemma 2 in Section 5. Then, we prove Lemma 1 in Section 6.

5 Adaptive Security against the Type-II Adversary

Here, we repeat the definition of a Type-II adversary:

Type-II Adversary: \( A \) is called Type-II if it does not make secret key reveal queries on any \( ID \in \text{prefix}^+(ID^*) \).

We first provide an overview of Emura et al.’s proof for adaptively secure RHIBE schemes against Type-II Adversary [ETW20] and observe that a simple dual system translation cannot prove the adaptive security of our RHIBE scheme. Subsequently, we explain the proof of the adaptive security of our RHIBE scheme against the Type-II adversary.

**Overview of Emura et al.’s Proof [ETW20]**. Emura et al.’s proof is an adaptively secure adaptation of Seo–Emura’s proof for selectively secure RHIBE schemes [SE15]. Specifically, Emura
et al. reduced the adaptive security of the underlying HIBE scheme to the adaptive security of their proposed RHIBE schemes. Similar to our secret key $s_k_{T_0}$, Emura et al.’s secret key $s_k_{ID}$ consists of sub-secret keys $sk_{ID,\theta}$. Furthermore, similar to our sub-secret key $sk_{ID,\theta}$, Emura et al.’s sub-secret key $sk_{ID,\theta}$ is an HIBE secret key with $k_{pa(ID),\theta} \leftarrow_R \mathbb{Z}_p^{k+1}$ as the master secret key. Although Emura et al.’s key update $ku_{pa(ID),T}$ does not have a helper key update $ku_{pa(ID),T}$, our KGC’s key update $ku_{kgc,T}$, the former consists of sub-key updates $ku_{pa(ID),T}$.

Based on the modification, Emura et al. switched the position of $MSK$ so that their sub-secret key $sk_{ID,\theta}$ and sub-key update $ku_{pa(ID),T,\theta}$ are HIBE secret keys with $MSK - k_{pa(ID),\theta}$ and $k_{pa(ID),\theta}$, respectively, as the master secret key. This switching enables the reduction algorithm to answer all $sk_{ID,\theta}$ and $ku_{pa(ID),T,\theta}$ upon $A$’s queries. Specifically, since all $sk_{ID,\theta}$ that is revealed to $A$ satisfy $ID \notin prefix^+(ID^*)$, the reduction algorithm can interact with the HIBE challenger to receive an ID’s HIBE secret key that is sufficient for creating $sk_{ID,\theta}$. Since the reduction algorithm knows $k_{pa(ID),\theta}$, the algorithm can create all $ku_{pa(ID),T,\theta}$ by itself.

The last obstacle to overcome is answering $delk_{pa(ID),\theta}$ to $A$. The security proof of RHIBE without insider security is relatively easy since we can neglect the obstacle. When the delegation key $delk_{ID,\theta}$ follows the same distribution as our scheme, the above proof strategy fails unless the reduction algorithm knows the master secret key of the underlying HIBE scheme. To avoid the obstacle and achieve insider security, Emura et al. defined the delegation key $delk_{ID,\theta}$ so that it is an HIBE secret key with $k_{pa(ID),\theta} \leftarrow_R \mathbb{Z}_p^{k+1}$ as the master secret key. Then, based on the fact that $ID \notin prefix^+(ID^*)$, the reduction algorithm interacts with the HIBE challenger to receive an ID’s HIBE secret key that is sufficient for creating $sk_{ID,\theta}$. Therefore, Emura et al.’s delegation key $delk_{ID,\theta}$ becomes larger than that of ours by a factor $O(L - |ID|)$.

**Overview of Our Proof against the Type-II Adversary.** As we observed in Section 4, the task of our proof is changing all $ku_{kgc,T,\theta}$, $ku_{ID,T}$, and $dk_{ID,T}$ associated with $MSK$ to be semi-functional. Here, we observe that Emura et al.’s technique cannot prove the adaptive security of our scheme. When we switch the position of $MSK$, we do not have to change $ku_{kgc,T,\theta}$ and $ku_{ID,T}$ to be semi-functional. Instead, we have to change $delk_{ID,\theta}$ and $sk_{ID,\theta}$ associated with $MSK$ to be semi-functional. Since all $sk_{ID,\theta}$ that is revealed to $A$ satisfy $ID \notin prefix^+(ID^*)$, we can change all $sk_{ID,\theta}$ to be semi-functional by properly introducing the semi-functional distribution of $sk_{ID,\theta}$. On the other hand, we cannot change the distribution of $delk_{ID,\theta} = MSK - k_{ID,\theta}$ unless we use larger delegation keys as adopted in Emura et al.’s scheme.

To avoid this obstacle, we employ the dual system encryption methodology and another theoretic trick that we call the *semi-functional randomness switching*. This switching was implicitly introduced by Takayasu and Watanabe [TW21] to construct adaptively secure anonymous (non-hierarchical) RIBE schemes. In the following texts, we explain how Takayasu and Watanabe changed $ku_{kgc,T}$ to be semi-functional. Initially, Takayasu and Watanabe changed all $sk_{ID,\theta}$ to be semi-functional. Then, they guessed the value of $\star$ with a polynomial reduction loss $|T|$ and changed all $ku_{kgc,T}$ for $T \neq \star$ to be semi-functional. Finally, from these changes, they showed that normal and semi-functional $ku_{kgc,T}$ were identically distributed.

By following their argument and applying their strategy to the hierarchical case, we can prove the adaptive security of our RHIBE scheme. Before providing an overview of our proof, we introduce the following seed secret keys and its semi-functional distribution.

**Normal Seed Secret Keys:** A normal seed secret key is defined as $s.sk_{ID} := (s.SK_{ID,0}, s.SK_{ID,1}$,
\( \text{s.SK}_{\text{ID},2}, (\text{s.SK}_{\text{ID},\ell})_{\ell \in [|[\text{ID}] + 1, L]}): \)

\[
\begin{align*}
\text{s.SK}_{\text{ID},0} & := [\text{Zr}_{\text{ID}}]_2, \\
\text{s.SK}_{\text{ID},1} & := [(\text{V}_0 + \text{id}_1 \text{V}_1 + \cdots + \text{id}_{|\text{ID}|} \text{V}_{|\text{ID}|}) \text{Zr}_{\text{ID}}]_2, \\
\text{s.SK}_{\text{ID},2} & := [\text{V}_{L+2} \text{Zr}_{\text{ID}}]_2, \\
\tilde{\text{SK}}_{\text{ID},\ell} & := [\text{V}_\ell \text{Zr}_{\text{ID}}]_2
\end{align*}
\]

where \( \text{r}_{\text{ID}} \leftarrow_R \mathbb{Z}_p^k \).

**Semi-functional Seed Secret Keys:** A semi-functional seed secret key is defined as \( \text{s.sK}_{\text{ID}} := (\text{s.SK}_{\text{ID},0}, \text{s.SK}_{\text{ID},1}, \text{s.SK}_{\text{ID},2}, (\text{s.SK}_{\text{ID},\ell})_{\ell \in [|[\text{ID}] + 1, L]}): \)

\[
\begin{align*}
\text{s.SK}_{\text{ID},0} & := [\text{Zr}_{\text{ID}}]_2, \\
\text{s.SK}_{\text{ID},1} & := [\text{s} \cdot \text{a}^*]_2 \cdot [(\text{V}_0 + \text{id}_1 \text{V}_1 + \cdots + \text{id}_{|\text{ID}|} \text{V}_{|\text{ID}|}) \text{Zr}_{\text{ID}}]_2, \\
\text{s.SK}_{\text{ID},2} & := [\text{V}_{L+2} \text{Zr}_{\text{ID}}]_2, \\
\tilde{\text{SK}}_{\text{ID},\ell} & := [\text{V}_\ell \text{Zr}_{\text{ID}}]_2
\end{align*}
\]

where \( \text{r}_{\text{ID}} \leftarrow_R \mathbb{Z}_p^k \), and the semi-functional randomness \( \alpha \leftarrow_R \mathbb{Z}_p^* \) is shared with all seed secret keys unless otherwise stated. Here, the boxed part denotes the change from the normal seed secret key.

We prove the adaptive security of our RHIBE scheme against the Type-II adversary based on the following sequence of games:

**Game\( _{\text{II},0} \):** This is a real security game between the challenger \( \mathcal{C} \) and adversary \( \mathcal{A} \).

**Game\( _{\text{II},1} \):** This game is the same as **Game\( _{\text{II},0} \) except that the challenge ciphertext \( \text{ct}^* \) is semi-functional.

**Game\( _{\text{II},2} \):** This game is the same as **Game\( _{\text{II},1} \) except that \( \mathcal{C} \) modifies the method for creating secret keys \( \text{sk}_{\text{ID}} \), key updates \( \text{ku}_{\text{ID},T} \), and decryption keys \( \text{dk}_{\text{ID},T} \) as follows:

Secret Key Creation: Upon \( \mathcal{A} \)'s secret key generation queries on \( \text{ID} \), \( \mathcal{C} \) does not create sub-secret keys \( \text{sk}_{\text{ID},\theta} \). Upon \( \mathcal{A} \)'s secret key reveal queries on \( \text{ID} \), \( \mathcal{C} \) first creates a normal seed secret keys \( \text{s.sK}_{\text{ID}} \). Then, \( \mathcal{C} \) uses \( \text{s.sK}_{\text{ID}} \) to create all sub-secret keys \( \text{sk}_{\text{ID},\theta} \).

Key Update Creation: Upon \( \mathcal{A} \)'s secret key generation queries, \( \mathcal{C} \) first creates seed secret keys \( \text{s.sK}_{\text{ID}}^{(2)} \). \( \mathcal{C} \) creates \( \text{ku}_{\text{kgc},T} \) in the same way as done in the real scheme. To create \( \text{ku}_{\text{ID},T} \) such that \( |\text{ID}| \geq 1 \), \( \mathcal{C} \) creates sub-key updates \( \text{ku}_{\text{ID},T,\theta} \) in the same way as done in the real scheme, while \( \mathcal{C} \) uses \( \text{s.sK}_{\text{ID}}^{(2)} \) to create all helper key updates \( \text{ku}_{\text{ID},T} \).

Decryption Key Creation: Upon \( \mathcal{A} \)'s decryption key reveal queries, \( \mathcal{C} \) does not use \( \text{sk}_{\text{ID}} \) and \( \text{ku}_{\text{pa}(\text{ID}),T} \) to create \( \text{dk}_{\text{ID},T} \).

**Game\( _{\text{II},3} \):** This game is the same as **Game\( _{\text{II},2} \) except that \( \mathcal{C} \) creates semi-functional seed secret keys \( \text{s.sK}_{\text{ID}}^{(1)} \) upon \( \mathcal{A} \)'s secret key reveal queries.

**Game\( _{\text{II},4} \):** This game is the same as **Game\( _{\text{II},3} \) except that \( \mathcal{C} \) always creates semi-functional \( \text{ku}_{\text{kgc},T} \).

**Game\( _{\text{II},5} \):** This game is the same as **Game\( _{\text{II},4} \) except that \( \mathcal{C} \) always creates semi-functional helper key updates \( \text{ku}_{\text{ID},T} \) to create \( \text{ku}_{\text{ID},T} \).

**Game\( _{\text{II},6} \):** This game is the same as **Game\( _{\text{II},5} \) except that \( \mathcal{C} \) creates semi-functional \( \text{dk}_{\text{ID},T} \) to answer \( \mathcal{A} \)'s decryption key reveal queries.

**Game\( _{\text{II},7} \):** This game is the same as **Game\( _{\text{II},6} \) except that the challenge ciphertext \( \text{ct}^* \) is the semi-functional encryption of a random plaintext.
Table 1: Distributions of ct\textsuperscript{*}, s.sk\textsubscript{ID}\textsuperscript{(1)} for creating sk\textsubscript{ID,θ}, and sk\textsubscript{ID,θ} in each game in the proof against the Type-II adversary. In the column ct\textsuperscript{*}, we specify the distribution and encrypted plaintext. In the other columns, we specify the distributions and semi-functional randomness of s.sk\textsubscript{ID} and sk\textsubscript{ID,θ}.

<table>
<thead>
<tr>
<th>Game</th>
<th>ct\textsuperscript{*}</th>
<th>s.sk\textsubscript{ID}\textsuperscript{(1)}</th>
<th>sk\textsubscript{ID,θ}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Game\textsubscript{II},0</td>
<td>normal</td>
<td>normal</td>
<td>normal</td>
</tr>
<tr>
<td>M\textsubscript{coin}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Game\textsubscript{II},1</td>
<td>semi-functional</td>
<td>normal</td>
<td>normal</td>
</tr>
<tr>
<td>M\textsubscript{coin}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Game\textsubscript{II},2</td>
<td>semi-functional</td>
<td>normal</td>
<td>normal</td>
</tr>
<tr>
<td>M\textsubscript{coin}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Game\textsubscript{II},3</td>
<td>semi-functional</td>
<td>semi-functional</td>
<td>semi-functional</td>
</tr>
<tr>
<td>M\textsubscript{coin}</td>
<td>α ← R ∗ \mathbb{Z}_p</td>
<td>\tilde{r}<em>{ID,θ}α ; \tilde{r}</em>{ID,θ} ← R \mathbb{Z}_p</td>
<td></td>
</tr>
<tr>
<td>Game\textsubscript{II},4</td>
<td>semi-functional</td>
<td>semi-functional</td>
<td>semi-functional</td>
</tr>
<tr>
<td>M\textsubscript{coin}</td>
<td>α ← R ∗ \mathbb{Z}_p</td>
<td>\tilde{r}<em>{ID,θ}α ; \tilde{r}</em>{ID,θ} ← R \mathbb{Z}_p</td>
<td></td>
</tr>
<tr>
<td>Game\textsubscript{II},5</td>
<td>semi-functional</td>
<td>semi-functional</td>
<td>semi-functional</td>
</tr>
<tr>
<td>M\textsubscript{coin}</td>
<td>α ← R ∗ \mathbb{Z}_p</td>
<td>\tilde{r}<em>{ID,θ}α ; \tilde{r}</em>{ID,θ} ← R \mathbb{Z}_p</td>
<td></td>
</tr>
<tr>
<td>Game\textsubscript{II},6</td>
<td>semi-functional</td>
<td>semi-functional</td>
<td>semi-functional</td>
</tr>
<tr>
<td>M\textsubscript{coin}</td>
<td>α ← R ∗ \mathbb{Z}_p</td>
<td>\tilde{r}<em>{ID,θ}α ; \tilde{r}</em>{ID,θ} ← R \mathbb{Z}_p</td>
<td></td>
</tr>
<tr>
<td>Game\textsubscript{II},7</td>
<td>semi-functional</td>
<td>semi-functional</td>
<td>semi-functional</td>
</tr>
<tr>
<td>M\textsubscript{∗} ← \mathbb{G}_T</td>
<td>α ← R ∗ \mathbb{Z}_p</td>
<td>\tilde{r}<em>{ID,θ}α ; \tilde{r}</em>{ID,θ} ← R \mathbb{Z}_p</td>
<td></td>
</tr>
</tbody>
</table>
Table 2: Distributions of \( ku_{k_{gc,T}, \theta} \), \( \overline{ku}_{ID,T} \) for \(|ID| \geq 1 \), and \( dk_{ID,T} \) in each game in the proof against the Type-II adversary. We specify the distributions and semi-functional randomness of \( ku_{k_{gc,T}, \theta} \), \( \overline{ku}_{ID,T} \), and \( dk_{ID,T} \).

<table>
<thead>
<tr>
<th>Game</th>
<th>( ku_{k_{gc,T}, \theta} )</th>
<th>( \overline{ku}_{ID,T} )</th>
<th>( dk_{ID,T} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( Game_{II,0} )</td>
<td>normal</td>
<td>normal</td>
<td>normal</td>
</tr>
<tr>
<td>( Game_{II,1} )</td>
<td>normal</td>
<td>normal</td>
<td>normal</td>
</tr>
<tr>
<td>( Game_{II,2} )</td>
<td>normal</td>
<td>normal</td>
<td>normal</td>
</tr>
<tr>
<td>( Game_{II,3} )</td>
<td>normal</td>
<td>normal</td>
<td>normal</td>
</tr>
<tr>
<td>( Game_{II,4} )</td>
<td>semi-functional</td>
<td>normal</td>
<td>normal</td>
</tr>
<tr>
<td>( \alpha \leftarrow R \mathbb{Z}_p^* )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( Game_{II,5} )</td>
<td>semi-functional</td>
<td>semi-functional</td>
<td>normal</td>
</tr>
<tr>
<td>( \alpha \leftarrow R \mathbb{Z}_p^* ) &amp; ( \alpha \leftarrow R \mathbb{Z}_p^* ) &amp;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( Game_{II,6} )</td>
<td>semi-functional</td>
<td>semi-functional</td>
<td>semi-functional</td>
</tr>
<tr>
<td>( \alpha \leftarrow R \mathbb{Z}_p^* ) &amp; ( \alpha \leftarrow R \mathbb{Z}_p^* ) &amp; ( \alpha \leftarrow R \mathbb{Z}_p^* )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( Game_{II,7} )</td>
<td>semi-functional</td>
<td>semi-functional</td>
<td>semi-functional</td>
</tr>
<tr>
<td>( \alpha \leftarrow R \mathbb{Z}_p^* ) &amp; ( \alpha \leftarrow R \mathbb{Z}_p^* ) &amp; ( \alpha \leftarrow R \mathbb{Z}_p^* )</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In Tables 1 and 2, we summarize the distributions of \( ct^* \), \( sk_{ID} \), \( ku_{k_{gc,T}} \), \( ku_{ID,T} \), and \( dk_{ID,T} \) in each game. \( Game_{II,0} \) is the real security game. In \( Game_{II,1} \), we change the challenge ciphertext \( ct^* \) to be semi-functional as per the standard dual system argument (Lemma 3). \( Game_{II,2} \) is the conceptual change that is useful to reduce the reduction loss. Thus, the indistinguishability \( Game_{II,1} \equiv Game_{II,2} \) (Lemma 4) immediately holds. In \( Game_{II,2} \), \( C \) does not create \( sk_{ID, \theta} \) and \( ku_{ID,T} \) as the real scheme. In turn, \( C \) first creates seed secret keys \( s.sk_{ID}^{(1)} \) and \( s.sk_{ID}^{(2)} \), and uses the seed secret keys to create \( sk_{ID, \theta} \) and \( \overline{ku}_{ID,T} \), respectively. In \( Game_{II,3} \), all \( s.sk_{ID}^{(1)} \) revealed to \( A \) become semi-functional. We use the standard dual system argument to prove the indistinguishability \( Game_{II,2} \approx_c Game_{II,3} \) (Lemma 5) by considering the fact that \( ID \notin \text{prefix}^+(ID^*) \). Then, we can apply the semi-functional randomness switching for \( ku_{k_{gc,T}, \theta} \). In \( Game_{II,4} \), all \( ku_{k_{gc,T}, \theta} \) become semi-functional. Thus, the proof of the indistinguishability \( Game_{II,3} \equiv Game_{II,4} \) (Lemma 9) is the first main part of the proof. Here, we use the following two facts:

- All \( sk_{ID, \theta} \) revealed to \( A \) such that \( pa(ID) = k_{gc} \) are created by semi-functional \( s.sk_{ID}^{(1)} \).
- No \( delk_{k_{gc, \theta}} \) are revealed to \( A \).

Based on the facts, the randomness of \( k_{k_{gc, \theta}} \leftarrow R \mathbb{Z}_p^{k+1} \) enables us to prove that normal and semi-functional \( ku_{k_{gc,T}} \) are identically distributed.

In \( Game_{II,5} \), all \( \overline{ku}_{ID,T, \theta} \) such that \(|ID| \geq 1 \) are created by semi-functional \( s.sk_{ID}^{(2)} \). In other words, all \( \overline{ku}_{ID,T, \theta} \) follow the semi-functional distribution in \( Game_{II,5} \). The proof of the indistinguishability \( Game_{II,4} \equiv Game_{II,5} \) (Lemma 10) is the second main part of the proof, and it is more technical than the proof of \( Game_{II,3} \equiv Game_{II,4} \). If \( ID \in \text{prefix}^+(ID^*) \), we cannot apply the standard dual system argument to change \( s.sk_{ID}^{(2)} \) to be semi-functional. In contrast, if \( ID \notin \text{prefix}^+(ID^*) \), we cannot apply
the semi-functional randomness switching since \( \text{del} k_{\text{ID}, \theta} \) may be revealed to \( A \) via secret key reveal queries. Thus, in brief, the proof is a combination of the standard dual system argument and semi-functional randomness switching. By following the same procedure, either the standard dual system argument or semi-functional randomness switching enables us to prove that normal and semi-functional \( sk_{\text{ID}, T} \) are identically distributed.

In Game\(_{\text{II},6}\), we change all \( dk_{\text{ID}, T} \) to be semi-functional one by one. Here, the standard dual system argument is sufficient for proving the indistinguishability Game\(_{\text{II},5} \approx_c \text{Game}\(_{\text{II},6}\) (Lemma 14) considering the fact that \((\text{ID}, T) \neq (\text{ID}' , T')\). Finally, in Game\(_{\text{II},7}\), we change the challenge ciphertext \( ct^* \) to be a semi-functional encryption of a random plaintext. Since all \( ku_{\text{gc}, T, \theta}, ku_{\text{ID}, T}, \) and \( dk_{\text{ID}, T} \) are semi-functional, the standard dual system argument is sufficient for proving the indistinguishability Game\(_{\text{II},6} \equiv \text{Game}\(_{\text{II},7}\) (Lemma 18).

### 5.1 Proof of Lemma 2

Now, we are ready to prove Lemma 2.

**Proof of Lemma 2.** Let \( \text{Adv}_i(\lambda) \) denote \( A \)'s advantage in Game\(_{\text{II},i}\). Hereafter, we prove that the difference of \( A \)'s advantage between each game (i.e., \( |\text{Adv}_{i-1}(\lambda) - \text{Adv}_i(\lambda)| \)) is negligible. The key points to note is the transitions Game\(_{\text{II},3} \equiv \text{Game}\(_{\text{II},4}\) and Game\(_{\text{II},4} \approx_c \text{Game}\(_{\text{II},5}\) since we have to change \( ku_{\text{gc}, T} \) and \( ku_{\text{ID}, T} \) such that \( \text{ID} \in \text{prefix}^+(\text{ID}') \land T = T^* \) to be semi-functional. In other words, we rely on Chen-Gong’s technique [CG17] to prove the other transitions.

**Lemma 3** (Ciphertext Invariance, Game\(_{\text{II},0} \approx_c \text{Game}\(_{\text{II},1}\)).

Game\(_{\text{II},0}\) and Game\(_{\text{II},1}\) are computationally indistinguishable under the MDDH assumption in \( G_1 \). Specifically, for any PPT adversary \( A \) making at most \( Q_{\text{gen}} \) secret key generation queries, there exists a reduction algorithm \( B_0 \) such that

\[
|\text{Adv}_0(\lambda) - \text{Adv}_1(\lambda)| \leq \text{Adv}_{\text{MDDH}}^{G_1}(\lambda)
\]

and \( T(B_0) \approx T(A) + k^2 Q_{\text{gen}} |T| \cdot \text{poly}(\lambda, L), \) where \( \text{poly}(\lambda, L) \) is independent of \( T(A) \).

The proof completely follows the same step of Chen-Gong [CG17]. For the completeness, we formally prove Lemma 3.

**Proof of Lemma 3.** The reduction algorithm \( B_0 \) is given an MDDH instance in \( G_1 \):

\[
(\mathcal{G}(1^\lambda), [A]_1, [c]_1 = [A + e]_1)
\]

where \( A \leftarrow D_k, s \leftarrow \mathbb{Z}_p^k, e = 0 \) or \( e \leftarrow \mathbb{Z}_p^{k+1} \). Then, \( B_0 \) samples random matrices \((V_\ell)_{\ell \in [0, L+2]}, Z) \leftarrow R (\mathbb{Z}_p^{k+1} \times \mathbb{Z}_p^{k \times k})^L \), and a random vector \( k \leftarrow R \mathbb{Z}_p^{k+1} \)

uniformly at random. \( B_0 \) then returns

\[
\text{MPK} = (\; [A]_1; (V_\ell^T A)_{\ell \in [0, L+2]}; Z_2; (V_\ell Z)_{\ell \in [0, L+2]}; [A^T k]_T)
\]

to \( A \). Since \( B_0 \) knows MSK = \( k \), it can answer all \( A \)'s key queries in the same way as the real scheme.

Upon \( A \)'s challenge query on \((\text{ID}', T', M_{01}', M_{02}')\), \( B_0 \) samples coin \( \leftarrow R \{0, 1\}, (v_0, v_1, \ldots, v_{|\text{ID}'|}, v_{L+1}) \leftarrow R \mathbb{Z}_p^{k+1}, \) and returns the challenge ciphertext \( ct^* = (C_0, C_1, C_1', C_2, \text{tag}, \text{tag}') \):

\[
\begin{align*}
tag &= v_0 + v_1 \text{ID}^*_v + \cdots + v_{|\text{ID}'|} \text{ID}^*_{|\text{ID}'|}, \quad \text{tag'} = v_0 + v_{L+1} T^*, \quad C_0 = [c]_1, \\
C_1 &= [(V_0 + \text{ID}^*_v V_1 + \cdots + \text{ID}^*_{|\text{ID}'|} V_{|\text{ID}'|} + \text{tag}V_{L+2})^T]_1, \\
C_1' &= [(V_0 + T^* V_{L+1} + \text{tag}V_{L+2})^T]_1, \quad C_2 = M_{\text{coin}}^* (c^T \text{MSK})_T,
\end{align*}
\]

to \( A \). If \( e = 0, ct^* \) is a normal ciphertext as in Game\(_{\text{II},0}\). Otherwise, \( ct^* \) is a semi-functional ciphertext as in Game\(_{\text{II},1}\). Thus, we complete the proof. \( \square \)
Lemma 4 (GameII,1 ≡ GameII,2). GameII,1 and GameII,2 are identically distributed from A’s view. Specifically, for any PPT Type-II adversary A, it holds that

\[ \text{Adv}_{II,1}(\lambda) = \text{Adv}_{II,2}(\lambda). \]

The proof is clear since the key creations of our scheme are path-oblivious.

Proof of Lemma 4. C creates MPK in the same way as the real scheme. Hereafter, we describe how C creates skID, kuID,T, dkID,T, and ct* in GameII,2.

Secret Key Creation: Upon A’s secret key generation query on ID, C runs \((\eta_{ID}, BT_{pa(ID)}) \leftarrow \text{CS.Assign}(BT_{pa(ID)}, ID)\) and performs the delegation key generation in the same way as the real scheme and runs BT_{ID} \leftarrow \text{CS.SetUp}(1^\lambda, ID).

Upon A’s secret key reveal query on ID, C samples \(r_{ID}^{(1)} \leftarrow R Z_p^k\) and creates a normal seed secret key \(s.skID = (s.SK_{ID,0}^{(1)}, s.SK_{ID,1}^{(1)}, s.SK_{ID,2}^{(1)}, \tilde{\sigma}(s.SK_{ID,1}^{(1)}))_{\ell \in [1, |ID| + 1, L]}\) by computing (8). Then, for each \(\theta \in \text{Path}(BT_{pa(ID)}, \eta_{ID})\), C retrieves the delegation key \(delk_{pa(ID), \theta} = k_{pa(ID), \theta}\), samples \(\tilde{r}_{ID, \theta} \leftarrow R Z_p^k\), and \(\tilde{r}_{ID, \theta} \leftarrow R Z_p^k\), and computes a sub-secret key \(s.sk_{ID, \theta} = (SK_{ID, \theta,0}, SK_{ID, \theta,1}, SK_{ID, \theta,2}, \tilde{SK}_{ID, \theta, \ell})_{\ell \in [1, |ID| + 1, L]}\):

\[
\begin{align*}
SK_{ID, \theta, 0} &= (s.SK_{ID, 0}^{(1)})_{\tilde{r}_{ID, \theta}} \cdot |Z\tilde{r}_{ID, \theta}|_2, \\
SK_{ID, \theta, 1} &= [k_{pa(ID), \theta}]_2 \cdot (s.SK_{ID, 1}^{(1)})_{\tilde{r}_{ID, \theta}} \cdot [(V_0 + id_1 V_1 + \cdots + id_{|ID|} V_{|ID|}) Z\tilde{r}_{ID, \theta}]_2, \\
SK_{ID, \theta, 2} &= (s.SK_{ID, 2}^{(1)})_{\tilde{r}_{ID, \theta}} \cdot [V_{L+2} Z\tilde{r}_{ID, \theta}]_2, \\
\tilde{SK}_{ID, \theta, \ell} &= (s.SK_{ID, \ell}^{(1)})_{\tilde{r}_{ID, \theta}} \cdot [V_{|ID|} Z\tilde{r}_{ID, \theta}]_2.
\end{align*}
\]

The distribution is the same as in GameII,1 by setting \(r_{ID, \theta} = \tilde{r}_{ID, \theta} \cdot r_{ID}^{(1)} + \tilde{r}_{ID, \theta}\). Due to the fresh random \(\tilde{r}_{ID, \theta} \leftarrow R Z_p^k\), \(r_{ID, \theta}\) is distributed in \(Z_p^k\) uniformly at random.

Key Update Creation: Upon A’s secret key generation query on ID, C samples \(r_{ID}^{(2)} \leftarrow R Z_p^k\) and creates a normal seed secret key \(s.sk_{ID}^{(2)} = (s.SK_{ID,0}^{(2)}, s.SK_{ID,1}^{(2)}, s.SK_{ID,2}^{(2)}, \tilde{\sigma}(s.SK_{ID,1}^{(2)}))_{\ell \in [1, |ID| + 1, L]}\) by computing (8). C runs \(KU_{ID,T} \leftarrow \text{CS.Cover}(BT_{ID}, RL_{ID,T})\) and creates \(ku_{gkc,T}\) by computing

\[
\begin{align*}
KU_{gkc,T, \theta, 0} &= [Zt_{gkc,T, \theta}]_2, \\
KU_{gkc,T, \theta, 1} &= [k - delk_{gkc,T, \theta}]_2 \cdot [(V_0 + TV_{L+1}) Zt_{gkc,T, \theta}]_2, \\
KU_{gkc,T, \theta, 2} &= [V_{L+2} Zt_{gkc,T, \theta}]_2.
\end{align*}
\]

To create \(ku_{ID,T}\) such that \(|ID| \geq 1\), C samples the ephemeral delegation key \(dek_{ID,T} = k_{ID,T} \leftarrow R Z_p^{k+1}\) and creates the sub-key update \(ku_{ID,T, \theta}\) in the same way as the real scheme. Then, C retrieves the delegation key \(k_{ID, \theta}\) and ephemeral delegation key \(k_{ID,T}\), samples \(\tilde{t}_{ID,T}, \tilde{t}_{ID,T}^{(1)} \leftarrow R Z_p^k\), and computes a helper key update \(\tilde{ku}_{ID,T} = (\tilde{KU}_{ID,T,0}, \tilde{KU}_{ID,T,1}, \tilde{KU}_{ID,T,2}, \tilde{KU}_{ID,T,3}, (\tilde{KU}_{ID,T, \ell})_{\ell \in [1, |ID| + 1, L]}):\)

\[
\begin{align*}
\tilde{KU}_{ID,T,0} &= s.SK_{ID, 0}^{(2)} \cdot [Z\tilde{t}_{ID,T}]_2, \\
\tilde{KU}_{ID,T,1} &= (k + k_{ID,T})_2 \cdot s.SK_{ID, 1}^{(2)} \cdot [(V_0 + id_1 V_1 + \cdots + id_{|ID|} V_{|ID|}) Z\tilde{t}_{ID,T}]_2 \\
&\quad \cdot [(V_0 + TV_{L+1}) Z\tilde{t}_{ID,T}]_2, \\
\tilde{KU}_{ID,T,2} &= s.SK_{ID, 2}^{(2)} \cdot [V_{L+2} Z\tilde{t}_{ID,T}]_2, \\
\tilde{KU}_{ID,T, \ell} &= s.SK_{ID, \ell}^{(2)} \cdot [V_{|ID|} Z\tilde{t}_{ID,T}]_2.
\end{align*}
\]
This is the normal helper key update as in Game$_{\Pi,1}$ by setting $t_{ID,T}^{(2)} = r_{ID}^{(2)} + \overline{t}_{ID,T}$. Due to the fresh random $\overline{t}_{ID,T} \leftarrow Z_p^k$, $\overline{t}_{ID,T}$ is distributed in $Z_p^k$ uniformly at random.

**Decryption Key Creations:** To create $dk_{ID,T}$, $C$ retrieves the master secret key $k$, samples $u_{ID,T}, u'_{ID,T} \leftarrow Z_p^k$, and computes $dk_{ID,T} = (DK_{ID,T,0}, DK'_{ID,T,0}, DK_{ID,T,1}, DK_{ID,T,2}, DK'_{ID,T,2})$:

\[
\begin{align*}
DK_{ID,T,0} &= [Zu_{ID,T}]_2, \\
DK'_{ID,T,0} &= [Zu'_{ID,T}]_2, \\
DK_{ID,T,1} &= [k]_2 \cdot [(V_0 + id_1 V_1 + \cdots + id_{|ID|} V_{|ID|}) Zu_{ID,T}]_2 \cdot [(V_0 + TV_{L+1}) Zu'_{ID,T}]_2, \\
DK_{ID,T,2} &= [V_{L+2} Zu_{ID,T}]_2, \\
DK'_{ID,T,2} &= [V_{L+2} Zu'_{ID,T}]_2, 
\end{align*}
\]

where the distribution is the same as the real scheme.

**Challenge Ciphertext Creation:** Upon $A$’s challenge query on $(ID^*, T^*, M_0^*, M_1^*)$, $C$ retrieves $(V_\ell)_{\ell \in [0,|ID^*|] \cup \{L+1\}}$ and master secret key $k$, samples $c \leftarrow Z_p^{k+1}$, $(v_0, v_1, \ldots, v_{|ID^*|}, v_{L+1}) \leftarrow Z_p^{|ID^*|+2}$, and coin $\leftarrow \{0,1\}$, and creates the semi-functional challenge ciphertext $ct^* = (\text{tag}, \text{tag}', C_0, C_1, C_1', C_2)$ by computing (4).

As we observed so far, all the elements distribute in the same way as in Game$_{\Pi,1}$. Thus, we complete the proof of Lemma 4. □

**Lemma 5 (Secret Key Invariance, Game$_{\Pi,2} \approx_c$ Game$_{\Pi,3}$).** Game$_{\Pi,2}$ and Game$_{\Pi,3}$ are computationally indistinguishable under the MDDH assumption in $G_2$. Specifically, for any PPT Type-II adversary $A$ making at most $Q_{\text{gen}}$ secret key generation queries and $Q_{\text{rev}}$ secret key reveal queries, there exist reduction algorithms $B_{\Pi,1}$ and $B_{\Pi,2}$ such that

\[
|\text{Adv}_{\Pi,2}(\lambda) - \text{Adv}_{\Pi,3}(\lambda)| \leq Q_{\text{rev}} \cdot \sum_{j \in [2]} \text{Adv}_{B_{\Pi,j}}^{\text{MDDH}-G_2}(\lambda) + \frac{4Q_{\text{rev}}}{p-1}
\]

and $\max_{j \in [2]} T(B_{\Pi,j}) \approx T(A) + k^2Q_{\text{gen}}|T| \cdot \text{poly}(\lambda, L)$, where $\text{poly}(\lambda, L)$ is independent of $T(A)$.

Here, we change each seed secret key $s.sk_{ID}^{(1)}$ on which $A$ makes a secret key reveal query to be semi-functional one by one. The proof essentially follows the standard dual system argument [CGW15, CG17, CW14]. For the completeness, we formally prove Lemma 5.

**Proof of Lemma 5.** To prove Lemma 5, we further introduce the following auxiliary distributions for seed secret keys.

**Pseudo-normal Seed Secret Keys:** A pseudo-normal seed secret key $s.sk_{ID} = (s.SK_{ID,0}, s.SK_{ID,1}, s.SK_{ID,2}, (s.SK_{ID,\ell})_{\ell \in [|ID|+1,L]})$ is defined as follows:

\[
\begin{align*}
s.SK_{ID,0} &= [Zr_{ID}]_2, \\
s.SK_{ID,1} &= [(V_0 + id_1 V_1 + \cdots + id_{|ID|} V_{|ID|}) Zr_{ID}]_2 \cdot [\hat{r} a_{1/2}^{1/v_0 + v_1 id_1 + \cdots + v_{|ID|} id_{|ID|}]}_2, \\
s.SK_{ID,2} &= [V_{L+2} Zr_{ID}]_2 \cdot [\hat{r} a_{1/2}^{1/v}]_2, \\
s.SK_{ID,\ell} &= [V_{\ell} Zr_{ID}]_2 \cdot [\hat{r} a_{1/2}^{1/v}]_2
\end{align*}
\]

where $r_{ID} \leftarrow Z_p^k$, $\hat{r} \leftarrow Z_p^k$, and $(v_0, v_1, \ldots, v_{|ID|}) \leftarrow Z_p^{|ID|+1}$ is the random coin used to create the challenge ciphertext. Here, the boxed parts denote the change from the normal seed secret key.

**Pseudo-SF Seed Secret Keys:** A pseudo-SF seed secret key $s.sk_{ID} = (s.SK_{ID,0}, s.SK_{ID,1}, s.SK_{ID,2}, (s.SK_{ID,\ell})_{\ell \in [|ID|+1,L]})$ is defined as follows:

\[
s.SK_{ID,0} := [Zr_{ID}]_2,
\]
where \( \mathbf{r}_{ID} \leftarrow_R \mathbb{Z}_p^k, \mathbf{\hat{r}} \leftarrow_R \mathbb{Z}_p^* \), \((v_0, v_1, \ldots, v_{|ID|}) \leftarrow_R \mathbb{Z}_p^{|ID|+1}\) is the random coin used to create the challenge ciphertext, and \( \alpha \leftarrow_R \mathbb{Z}_p^* \) is shared by all seed secret keys. Here, the boxed parts denote the change from the pseudo-normal seed secret key.

We further introduce the following sequence of games for \( q \in [0, Q_{rev}]\):

**Game\(_{II,2,q,1}\):** This game is the same as Game\(_{II,2}\) except that
- \( C \) creates *semi-functional* \( s.sk^{(1)}_{ID} \) to answer \( A \)'s first \( q - 1 \) secret key reveal queries,
- \( C \) creates *pseudo-normal* \( s.sk^{(1)}_{ID} \) to answer \( A \)'s \( q \)-th secret key reveal query,
- \( C \) creates *normal* \( s.sk^{(1)}_{ID} \) to answer \( A \)'s last \( (Q_{rev} - q) \) secret key reveal queries.

**Game\(_{II,2,q,2}\):** This game is the same as Game\(_{II,2,q,1}\) except that
- \( C \) creates *pseudo-SF* \( s.sk^{(1)}_{ID} \) to answer \( A \)'s \( q \)-th secret key reveal query.

**Game\(_{II,2,q,3}\):** This game is the same as Game\(_{II,2,q,2}\) except that
- \( C \) creates *semi-functional* \( s.sk^{(1)}_{ID} \) to answer \( A \)'s \( q \)-th secret key reveal query.

<table>
<thead>
<tr>
<th>Game</th>
<th>first ( q - 1 )</th>
<th>( q )-th</th>
<th>last ( Q_{rev} - q )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( s.sk^{(1)}_{ID} )</td>
<td>( s.sk^{(1)}_{ID} )</td>
<td>( s.sk^{(1)}_{ID} )</td>
</tr>
<tr>
<td>Game(_{II,2,q,1})</td>
<td>semi-functional</td>
<td>pseudo-normal</td>
<td>normal</td>
</tr>
<tr>
<td>Game(_{II,2,q,2})</td>
<td>semi-functional</td>
<td>pseudo-SF</td>
<td>normal</td>
</tr>
<tr>
<td>Game(_{II,2,q,3})</td>
<td>semi-functional</td>
<td>semi-functional</td>
<td>normal</td>
</tr>
</tbody>
</table>

In Table 3, we summarize the distributions of \( s.sk^{(1)}_{ID} \) in each game. By definition, Game\(_{II,2,0,3}\) = Game\(_{II,2}\) and Game\(_{II,2,Q_{rev},3}\) = Game\(_{II,3}\). Hereafter, we prove

\[
\text{Game}_{II,2,q-1,3} \approx_c \text{Game}_{II,2,q,1} \equiv \text{Game}_{II,2,q,2} \approx_c \text{Game}_{II,2,q,3},
\]

where the fact implies that Game\(_{II,2} \approx_c \text{Game}_{II,3}^\ast\).

**Lemma 6**: (Seed Secret Key Transition from Normal to Pseudo-normal, Game\(_{II,2,q-1,3} \approx_c \text{Game}_{II,2,q,1}\)). Game\(_{II,2,q-1,3}\) and Game\(_{II,2,q,1}\) are computationally indistinguishable under the MDDH assumption in \( \mathbb{G}_2 \). Specifically, for any PPT Type-II adversary \( A \) making at most \( Q_{gen} \) secret key generation queries, there exists a reduction algorithm \( B_{II,1} \) such that

\[
|\text{Adv}_{II,2,q-1,3}(\lambda) - \text{Adv}_{II,2,q,1}(\lambda)| \leq \text{Adv}_{B_{II,1}}^{\text{MDDH-\mathbb{G}_2}}(\lambda) + \frac{2}{p - 1}
\]

and \( T(B_{II,1}) \approx T(A) + k^2 Q_{gen}|T| \cdot \text{poly}(\lambda, L) \), where \( \text{poly}(\lambda, L) \) is independent of \( T(A) \).
Proof of Lemma 6. The reduction algorithm $B_{ID}$ is given an MDDH instance in $G_2$: $(G(1^λ), [b]_2, [b]_2 = [Br + ϵr]_2)$, where $B ← R D_k, r ← R Z_p^k, ϵ ← R Z_p^*$, and $e = (0, . . . , 0, 1)^T ∈ Z_p^{k+1}$. Hereafter, we assume that $r ← R Z_p^*$ in the latter case with the statistical difference $1/p$.

We describe how $C$ creates MPK, $sk_{ID}$, $ku_{ID,T}$, $dk_{ID,T}$, and $ct^*$.

MPK Creation: At the beginning of the game, $B_{ID}$ samples $(A, a^+) ← D_k$, $((V_ℓ)_{ℓ∈[0,L+2]}, Z) ← R (Z_p^{k+1} × Z_p^{k+1})^{L+3} × Z_p^{k×k}$, $b ← R Z_p^k$, $(v_0, v_1, . . . , v_{L+1}) ← R Z_p^{L+2}$, and $α ← R Z_p^*$. As the special case, we set $v_{L+2} = −1$. Let $B ∈ Z_p^{k×k}$ and $B' ∈ Z_p^{k×k}$ denote a top $k × k$ sub-matrix and bottom row vector of $B$, respectively, where $B$ is full-rank. Let

$$M := a^+(B'B)^{-1} ∈ Z_p^{k+1} × k$$

denote a matrix that is not computable by $B_{ID}$. $B_{ID}$ sets

$$V_ℓ = \tilde{V}_ℓ + v_ℓM, \quad [Z]_2 = [BZ]_2.$$  

Since $B$ is full-rank, $Z$ is distributed in $Z_p^{k×k}$ uniformly at random as required. Then, $B_{ID}$ computes

$$[\tilde{V}_ℓ^T A]_1 = [V_ℓ^T A - v_ℓ(BB)^{-1}^T \cdot (a^{+T} A)]_1 = [V_ℓ^T A]_1,$$

$$[\tilde{V}_ℓ BZ + v_ℓ a^+ BZ]_2 = [V_ℓ Z - v_ℓ a^+ (BB)^{-1} \cdot BZ + v_ℓ a^+ BZ]_2 = [V_ℓ Z]_2,$$

for $ℓ ∈ [0, L + 2]$. Therefore, $B_{ID}$ can compute

$$MPK = ([A]_1, ([V_ℓ^T A]_1)_{ℓ∈[0,L+2]}, [Z]_2, ([V_ℓ Z]_2)_{ℓ∈[0,L+2]}, [A^+ k]_T)$$

that is distributed in the same way as the real scheme.

Secret Key Creation: Upon $A$’s secret key reveal queries on $ID$, $B_{ID}$ creates seed secret keys $s.sk_{ID}$ as follows:

- If this is not still the $q$-th query, $B_{ID}$ retrieves a delegation key $delk_{pa(ID),θ} = k_{pa(ID),θ}$ and $α$, samples $T_{ID}^{(1)} ← R Z_p^k$, and computes a semi-functional seed secret key $s.sk_{ID}^{(1)}$ by computing (9).

- If this is the $q$-th query, let $\tilde{B} ∈ Z_p^k$ and $b ∈ Z_p$ denote the first $k$-entries and last entry of $B$, respectively. Then, $B_{ID}$ retrieves $(v_0, v_1, . . . , v_{ID})$ and computes a seed secret key $s.sk_{ID}^{(1)}(s Sk_{ID,0}, s Sk_{ID,1}, s Sk_{ID,2}, (s Sk_{ID,ℓ})_{ℓ∈[ID]+1,L}):$

$$s.sk_{ID,0} = [\tilde{B}]_2$$

$s.sk_{ID,1}^{(1)} = [(\tilde{V}_0 + i_d \tilde{V}_1 + i_d [ID] \tilde{V}_{[ID]})(\tilde{B}]_2 \cdot [a^+ b^0 v_1 + i_d v_1 + \cdots + i_d [ID] v_{ID}], (14)$

$s.sk_{ID,2}^{(1)} = [\tilde{V}_{L+2} \tilde{B}]_2 \cdot [−a^+ b]_2, \quad s.sk_{ID,ℓ}^{(1)} = [\tilde{V}_ℓ \tilde{B}]_2 \cdot [a^+ b]_2.$

- If this is after the $q$-th query, $B_{ID}$ creates a normal seed secret key $s.sk_{ID}^{(1)}$ by computing (8). Then, $B_{ID}$ creates sub-secret keys $sk_{ID,θ}$ by computing (10).

Here, we check that the $q$-th queried seed secret key $s.sk_{ID}^{(1)}$ is properly distributed. By definition,

$$\tilde{B} = Br, \quad b = Br + ϵ,$$
where \( r \leftarrow_R \mathbb{Z}_p^k \), \( \hat{r} = 0 \) or \( \hat{r} \leftarrow_R \mathbb{Z}_p^k \). At first, we show that \( SK_{ID,0}^{(1)} \) is properly distributed by setting

\[
I_{ID}^{(1)} = Z^{-1} B = (\hat{Z}^{-1} B^{-1}) \cdot (Br) = \hat{Z}^{-1} r.
\]

Since \( \hat{Z} \) is distributed in \( \mathbb{Z}_p^{k 	imes k} \) uniformly at random, the matrix is full-rank with probability at least \( 1 - 1/(p - 1) \). Since \( r \) is distributed in \( \mathbb{Z}_p^k \) uniformly at random, \( I_{ID}^{(1)} \) also follows the same distribution if \( \hat{Z} \) is full-rank. Next, we observe that

\[
\tilde{\ell} B + v_e a^\perp b = (V_\ell - v_e a^\perp (BB^{-1})) \cdot (Br) + v_e a^\perp (Br + \hat{r}) \]

\[
= V_i Z r_{ID}^{(1)} + v_e \hat{r} a^\perp
\]

for \( \ell \in \{0, L + 2\} \), where \( v_{L+2} = -1 \). Therefore, it holds that

\[
sSK_{ID,1}^{(1)} = [(V_0 + \text{id}_1 V_1 + \text{id}_1^{\text{ID}} V_{\text{ID}})(Br)]_{2} \cdot [\hat{r} a^\perp]_{2} \cdot [\text{id}_1^{\text{ID}} v_0 + \text{id}_1^{\text{ID}} v_1 + \cdots + \text{id}_1^{\text{ID}} v_{\text{ID}},]
\]

\[
SK_{ID,2}^{(1)} = [V_{L+2} Z r_{ID}^{(1)}]_{2} \cdot [\hat{r} a^\perp]_{2}^{-1},
\]

\[
\overline{SK}_{ID,2}^{(1)} = [V Z r_{ID}^{(1)}]_{2} \cdot [\hat{r} a^\perp]_{2}.
\]

If \( \hat{r} = 0 \), \( sSK_{ID}^{(1)} \) is a normal seed secret key as in Game\( 1_{2,0,1} \). If \( \hat{r} \leftarrow_R \mathbb{Z}_p^k \), \( sSK_{ID}^{(1)} \) is a pseudo-normal seed secret key as in Game\( 1_{2,0,1} \).

**Key Update Creation:** \( B_{\text{ID},1} \) creates all \( ku_{\text{ID,T}} \) in the same way as in Game\( 2_{1,2} \).

**Decryption Key Creation:** \( B_{\text{ID},1} \) creates all \( dk_{\text{ID,T}} \) in the same way as in Game\( 2_{2,2} \).

**Challenge Ciphertext Creation:** Upon \( A \)'s challenge query on \( (\text{ID}^*, T^*, M_0^*, M_1^*) \), \( B_{\text{ID},1} \) retrieves \( (\tilde{V}_\ell)_{\ell \in \{0,L+2\}} \) and \( (v_0, v_1, \ldots, v_{\text{ID}}, v_{L+1}) \), samples \( c \leftarrow_R \mathbb{Z}_p^{k+1} \) and \( \text{coin} \leftarrow_R \{0, 1\} \), and creates the challenge ciphertext \( c^* = (\text{tag, tag}', C_0, C_1, C_1', C_2) \):

\[
\text{tag} = v_0 + v_1 \text{id}_1^\ast + \cdots + v_{\text{ID}} \text{id}_{\text{ID}}^\ast, \quad \text{tag}' = v_0 + v_{L+1} T^*,
\]

\[
C_0 = [c]_1,
\]

\[
C_1 = [(\tilde{V}_0 + \text{id}_1^\ast \tilde{V}_1 + \cdots + \text{id}_{\text{ID}}^\ast \tilde{V}_{\text{ID}} + \text{tag} \tilde{V}_{L+2})^\top c]_1
\]

\[
C_1' = [(\tilde{V}_0 + T^* \tilde{V}_{L+1} + \text{tag} \tilde{V}_{L+2})^\top c]_1, \quad C_2 = M_{\text{coin}} \cdot [c^\top k]^T.
\]

It is clear that \( \text{tag, tag}', C_0, C_2 \) are properly distributed. We check that \( C_1 \) and \( C_1' \) are also properly distributed as follows:

\[
C_1 = [(\tilde{V}_0 + \text{id}_1^\ast \tilde{V}_1 + \cdots + \text{id}_{\text{ID}}^\ast \tilde{V}_{\text{ID}} + \text{tag} \tilde{V}_{L+2})^\top c]_1
\]

\[
= [(V_0 - v_0 M) + \text{id}_1^\ast (V_1 - v_1 M) + \cdots + \text{id}_{\text{ID}}^\ast (V_{\text{ID}} - v_{\text{ID}} M)]^\top c]_1
\]

\[
= [(V_{L+2} + M)^\top c]_1 \cdot [v_0 + v_1 \text{id}_1^\ast + \cdots + v_{\text{ID}} \text{id}_{\text{ID}}^\ast] v_{\text{ID}},]
\]

\[
C_1' = [(\tilde{V}_0 + T^* \tilde{V}_{L+1} + \text{tag} \tilde{V}_{L+2})^\top c]_1
\]

\[
= [(V_0 - v_0 M) + T^*(V_{L+1} - v_{L+1} M)]^\top c]_1 \cdot [(V_{L+2} + M)^\top c]_1 \cdot [v_0 + v_{L+1} T^*]
\]

\[
= [(V_0 + T^* V_{L+1} + \text{tag} V_{L+2})^\top c]_1.
\]

Therefore, \( c^* \) is properly distributed semi-functional ciphertext.

Thus, we complete the proof of Lemma 6. \(\square\)
Lemma 7 (Seed Secret Key Transition from Pseudo-normal to Pseudo-SF,
Game\textsubscript{II,2,q,1} \equiv Game\textsubscript{II,2,q,2}). Game\textsubscript{II,2,q,1} and Game\textsubscript{II,2,q,2} are identically distributed. Specifically, for any Type-II adversary \( A \), it holds that
\[
\text{Adv}_{\text{II,2,q,1}}(\lambda) = \text{Adv}_{\text{II,2,q,2}}(\lambda). 
\]

Proof of Lemma 7. Here, we prove a stronger claim that Game\textsubscript{II,2,q,1} and Game\textsubscript{II,2,q,2} are identically distributed for any fixed
\[
\begin{align*}
&\bullet (A, a) \leftarrow R \mathcal{D}_k, \\
&\bullet ((V_\ell)_{\ell \in [0, L+2]}, Z) \leftarrow R (\mathbb{Z}_p^{(k+1) \times k})^{L+3} \times \mathbb{Z}_p^{k \times k}, \\
&\bullet \text{master secret key } k \leftarrow R \mathbb{Z}_p^{k+1}, \\
&\bullet c \leftarrow R \mathbb{Z}_p^{k+1} \text{ for creating the challenge ciphertext,} \\
&\bullet (ID^*, T^*, M_0^*, M_1^*) \in \mathbb{Z}_p^2 \times \mathcal{M}^2 \text{ and random coin } coin \leftarrow R \{0, 1\}, \\
&\bullet \text{delegation keys } k_{ID, \theta} \leftarrow R \mathbb{Z}_p^{k+1}, \\
&\bullet r_{ID}^{(1)} \leftarrow R \mathbb{Z}_p^k \text{ and } \hat{r} \leftarrow R \mathbb{Z}_p \text{ for creating } q\text{-th queried s.sk}\text{\textsubscript{ID}^{(1)}}, \\
&\bullet \text{semi-functional randomness } \alpha \leftarrow R \mathbb{Z}_p^\ast. 
\end{align*}
\]
Specifically, the randomness of \((v_0, v_1, \ldots, v_{L+1}) \leftarrow R \mathbb{Z}_p^{L+2}\) enables us to prove the claim. Since s.sk\textsubscript{ID}\textsuperscript{(1)} which is not \(q\)-th queried ones, s.sk\textsubscript{ID}\textsuperscript{(2)} and dk\textsubscript{ID,T} are created in the same way in both Game\textsubscript{II,2,q,1} and Game\textsubscript{II,2,q,2}, and all the other elements have been already fixed, it is sufficient to show that
\[
\begin{align*}
\{ v_0 + v_1 id_1^* + \cdots + v_{|ID|} id_1^{|ID|}, v_0 + v_{L+1} T^* \} \\
\{ v_0 + v_1 id_1 + \cdots + v_{|ID|} id_{ID}, (v_\ell)_{\ell \in [|ID|+1, L]} \}
\end{align*}
\equiv
\begin{align*}
\{ v_0 + v_1 id_1^* + \cdots + v_{|ID|} id_1^{|ID|}, v_0 + v_{L+1} T^*, \\
\alpha/\hat{r} + v_0 + v_1 id_1 + \cdots + v_{|ID|} id_{ID}, (v_\ell)_{\ell \in [|ID|+1, L]} \},
\end{align*}
\]
where \((v_0, v_1, \ldots, v_{L+1}) \leftarrow R \mathbb{Z}_p^{L+2}\). Here, the first and second elements are \text{\text{tag.tag}}', and last element is the exponent of \([\tilde{r}a]^\bot\) of \(q\)-th queried \((SK_{ID,\theta,1}, (SK_{ID,\theta,\ell})_{\ell \in [|ID|+1, L]})\) in Game\textsubscript{II,2,q,1} and Game\textsubscript{II,2,q,2}, respectively. Due to the randomness of \(v_{L+1} \leftarrow R \mathbb{Z}_p\), the second element is distributed in \(\mathbb{Z}_p^2\) uniformly at random. Since \(ID \notin \text{prefix}^T(ID^*)\) holds due to the definition of the Type-II adversary, the first and last elements are distributed in \(\mathbb{Z}_p^2\) uniformly at random due to the randomness of \((v_0, v_1, \ldots, v_L) \leftarrow R \mathbb{Z}_p^{L+1}\). Thus, we complete the proof of Lemma 7.

Lemma 8 (Secret Key Transition from Pseudo-SF to Semi-functional, Game\textsubscript{II,2,q,2} \approx c Game\textsubscript{II,2,q,3}). Game\textsubscript{II,2,q,2} and Game\textsubscript{II,2,q,3} are computationally indistinguishable under the MDDH assumption in \(\mathbb{G}_2\). Specifically, for any PPT Type-II adversary \( A \) making at most \(Q_{gen}\) secret key generation queries, there exists a reduction algorithm \( B_{II,2} \) such that
\[
|\text{Adv}_{\text{II,2,q,2}}(\lambda) - \text{Adv}_{\text{II,2,q,3}}(\lambda)| \leq \text{Adv}_{\mathcal{B}_{II,2}}^{\text{MDDH-G}_2}(\lambda) + \frac{2}{p - 1}
\]
and \( T(B_{II,2}) \approx T(A) + k^2 Q_{gen} |T| \cdot \text{poly}(\lambda, L) \), where \(\text{poly}(\lambda, L)\) is independent of \(T(A)\).

We omit the detailed proof of Lemma 8 since it is almost the same as the proof of Lemma 6. The only difference is that \( B_{II,2} \) creates s.sk\textsubscript{ID,1} upon \( A \)'s \(q\)-th secret key reveal query by
\[
s.SK_{ID,1}^{(1)} = ([a^\bot]_2 \cdot (\tilde{V}_0 + id_1 \tilde{V}_1 + id_{|ID|} \tilde{V}_{|ID|})b_2 \cdot [a^\bot b^\bot]_2^{v_0 + id_1 v_1 + \cdots + id_{|ID|} v_{|ID|}}.
\]
where the boxed parts denote the changes from (14). If \( \hat{r} \leftarrow_R Z_p^* \), s.sk\(_{ID}^{(1)}\) is a pseudo-SF seed secret key as in Game\(_{II,2,q,2}\). If \( \hat{r} = 0 \), s.sk\(_{ID}^{(1)}\) is a semi-functional seed secret key as in Game\(_{II,2,q,3}\).

By combining Lemmata 6–8, we have

\[
|\text{Adv}_{II,2}(\lambda) - \text{Adv}_{II,3}(\lambda)| \\
\leq \sum_{q \in [Q_{\text{rev}}]} |\text{Adv}_{II,2,q-1,3}(\lambda) - \text{Adv}_{II,2,q,1}(\lambda)| + \sum_{q \in [Q_{\text{rev}}]} |\text{Adv}_{II,2,q,1}(\lambda) - \text{Adv}_{II,2,q,2}(\lambda)| \\
+ \sum_{q \in [Q_{\text{rev}}]} |\text{Adv}_{II,2,q,2}(\lambda) - \text{Adv}_{II,2,q,3}(\lambda)| \\
\leq Q_{\text{rev}} \cdot \sum_{j \in [2]} \text{Adv}_{B_{II,j}}^{\text{MDDH-G}^2}(\lambda) + \frac{4Q_{\text{rev}}}{p - 1}.
\]

Thus, we complete the proof of Lemma 5. \(\square\)

**Lemma 9** (Semi-functional Randomness Switching for KGC’s Key Updates, Game\(_{II,3} \equiv\) Game\(_{II,4}\)). **Game\(_{II,3}\)** and **Game\(_{II,4}\)** are identically distributed from A’s view. Specifically, for any Type-II adversary A, it holds that

**Proof of Lemma 9.** Here, we prove a stronger claim that **Game\(_{II,3}\)** and **Game\(_{II,4}\)** are identically distributed from A’s view for any fixed

- \((A,a) \leftarrow_R D_k,\)
- \(((V_i)_{i \in [0,L+2]}), (Z) \leftarrow_R (Z_p^{(k+1) \times k})^{L+3} \times Z_p^{k \times k},\)
- master secret key \(k \leftarrow_R Z_p^{k+1},\)
- \(r^{(1)}_{ID} \leftarrow_R Z_p^k\) for creating all s.sk\(_{ID}^{(1)}\) such that |ID| = 1,
- \(r_{ID,\theta} \leftarrow_R Z_p^k\) for creating all sk\(_{ID,\theta}\) such that |ID| = 1,
- \(t_{kgc,T,\theta} \leftarrow_R Z_p^k\) for creating all ku\(_{kgc,T}\),
- \(\alpha \leftarrow_R Z_p^*\)

Specifically, the randomnesses of all \(\hat{r}_{ID,\theta} \leftarrow_R Z_p\) such that |ID| = 1 and all delk\(_{kgc,\theta}\) enable us to prove the claim. We note that sk\(_{ID}\) such that |ID| ≥ 2, ku\(_{ID,T}\) such that |ID| ≥ 1, and dk\(_{ID,T}\) are created in the same way in both **Game\(_{II,3}\)** and **Game\(_{II,4}\)**. We further note that even when r\(_{ID,\theta} \leftarrow_R Z_p^k\) are fixed, (SK\(_{ID,\theta,0}, SK_{ID,\theta,2}\)) do not reveal the quantities of \(\hat{r}_{ID,\theta} \leftarrow_R Z_p\) since they are masked by \(\hat{r}_{ID,\theta} \leftarrow_R Z_p^k\). Since r\(_{ID,\theta}\) and t\(_{kgc,T,\theta}\) are fixed, sk\(_{ID,\theta}\) such that |ID| = 1 and ku\(_{kgc,T,\theta}\) are distributed in the same way in both **Game\(_{II,3}\)** and **Game\(_{II,4}\)** except SK\(_{ID,\theta,1}\) and KU\(_{kgc,T,\theta,1}\). In **Game\(_{II,3}\)**, SK\(_{ID,\theta,1}\) and KU\(_{kgc,T,\theta,1}\) are distributed as follows:

\[
\text{SK}_{ID,\theta,1} = [k_{kgc,\theta} + \hat{r}_{ID,\theta} \alpha^2] \cdot ([V_0 + IDV_1]Zr_{ID,\theta})_2,
\]
Lemma 10 (Helper Key Update Invariance, Game_{II,4} \approx_c Game_{II,5}). Game_{II,4} and Game_{II,5} are computationally indistinguishable under the MDDH assumption in G_2. Specifically, for any PPT Type-I adversary A making at most Q_{gen} secret key generation queries, there exist reduction algorithms B_{II,3} and B_{II,4} such that
\[
|\text{Adv}_{II,4}(\lambda) - \text{Adv}_{II,5}(\lambda)| \leq Q_{gen} \cdot \sum_{j \in [2]} \text{Adv}_{B_{II,j+2}}^{MDDH-G_2}(\lambda) + \frac{4Q_{gen}}{p - 1}
\]
and max_{j \in [2]} T(B_{II,j+2}) \approx T(A) + k^2Q_{gen}|T| \cdot \text{poly}(\lambda, L), where \text{poly}(\lambda, L) is independent of T(A).

Let ID_q denote an identity on which A makes q-th secret key generation query. The structure of the proof may not look similar to the proof of Lemma 5, the spirit is almost the same.

Proof of Lemma 10. We further introduce the following sequence of games for q \in [0, Q_{gen}):

Game_{II,4,q,1}: This game is the same as Game_{II,4} except that
- If m < q, C always creates semi-functional s.sk_{ID,m}^{(2)},
- If m = q, C creates pseudo-normal s.sk_{ID_q}^{(2)},
- If m > q, C always creates normal s.sk_{ID_m}^{(2)}.

Game_{II,4,q,2}: This game is the same as Game_{II,4,q,1} except that
- If m = q, C creates pseudo-SF s.sk_{ID_q}^{(2)}.

Game_{II,4,q,3}: This game is the same as Game_{II,4,q,2} except that
- If m = q, C creates semi-functional s.sk_{ID_q}^{(2)}.

By definition, Game_{II,4,0,3} = Game_{II,4} and Game_{II,4,Q_{gen},3} = Game_{II,5}. Hereafter, we prove

Game_{II,4,q-1,3} \approx_c Game_{II,4,q,1} \equiv Game_{II,4,q,2} \approx_c Game_{II,4,q,3},

where the fact implies that Game_{II,4} \approx_c Game_{II,5}.

Lemma 11 (Sub-secret Key Transition from Normal to Pseudo-normal, Game_{II,4,q-1,3} \approx_c Game_{II,4,q,1}). Game_{II,4,q-1,3} and Game_{II,4,q,1} are computationally indistinguishable under the MDDH assumption in G_2. Specifically, for any PPT Type-II adversary A making at most Q_{gen} secret key generation queries, there exists a reduction algorithm B_{II,3} such that
\[
|\text{Adv}_{II,4,q-1,3}(\lambda) - \text{Adv}_{II,4,q,1}(\lambda)| \leq \text{Adv}_{B_{II,3}}^{MDDH-G_2}(\lambda) + \frac{2}{p - 1}
\]
and T(B_{II,3}) \approx T(A) + k^2Q_{gen}|T| \cdot \text{poly}(\lambda, L), where \text{poly}(\lambda, L) is independent of T(A).
We omit the proof of Lemma 11 since it is essentially the same as the proof of Lemma 6.

Lemma 12 (Sub-secret Key Transition from Pseudo-normal to Pseudo-SF, \(\text{Game}_{\Pi,4,q,1} \equiv \text{Game}_{\Pi,4,q,2}\)). Game_{\Pi,4,q,1} and Game_{\Pi,4,q,2} are identically distributed from \(\mathcal{A}\)'s view. Specifically, for any adversary Type-II \(\mathcal{A}\), it holds that

\[
\text{Adv}_{\Pi,4,q,1}(\lambda) = \text{Adv}_{\Pi,4,q,2}(\lambda).
\]

The proof of Lemma 12 is the second core part of the proof against the Type-II adversary since we have to change all \(s_sk^{(2)}_{\text{ID}}\) such that \(\text{ID} \in \text{prefix}^+(\text{ID}^*)\) to be semi-functional. Here, we use \(s_{\text{pa}(\text{ID})}\) to denote the \(q\)-th queried identity. When \(s_{\text{pa}(\text{ID})} \notin \text{prefix}^+(\text{ID}^*)\), we prove Lemma 12 in the same way as the proof of Lemma 7 by showing that pseudo-normal \(s_sk^{(2)}_{\text{pa}(\text{ID})}\) and pseudo-SF \(s_sk^{(2)}_{\text{pa}(\text{ID})}\) are identically distributed. On the other hand, when \(s_{\text{pa}(\text{ID})} \in \text{prefix}^+(\text{ID}^*)\), we cannot follow the dual system argument. Observe that what \(\text{Game}_{\Pi,4,q,2}\) does is essentially the same as the proof of Lemma 7.

Proof of Lemma 12. If \(\text{ID}_q = (\text{id}_{q,1}, \ldots, \text{id}_{q,|\text{ID}_q|}) \notin \text{prefix}^+(\text{ID}^*)\), we can show that pseudo-normal and pseudo-SF \(s_sk^{(2)}_{\text{ID}}\) are identically distributed by following the same argument as in the proof of Lemma 7. Then, in \(\text{Game}_{\Pi,4,q,1}\), \(\text{KU}_{\text{ID}_q,T,1}\) is distributed as follows:

\[
\text{KU}_{\text{ID}_q,T,1} = [k + k_{\text{ID}_q,T}]2 \cdot \lfloor \alpha a \rfloor \cdot \lfloor [V_0 + \text{id}_{q,1}V_1 + \cdots + \text{id}_{q,|\text{ID}_q|}V_{|\text{ID}_q|}]Zr_{\text{ID}_q,T} \rfloor^2 \\
\cdot \lfloor [V_0 + TV_{L+1}]Zr'_{\text{ID}_q,T} \rfloor^2 \cdot \lfloor \alpha a \rfloor^2.
\]

The distribution is the same as the distribution in \(\text{Game}_{\Pi,4,q,2}\).

If \(\text{ID}_q = (\text{id}_{q,1}, \ldots, \text{id}_{q,|\text{ID}_q|}) \in \text{prefix}^+(\text{ID}^*)\), we prove a stronger claim that \(\text{Game}_{\Pi,4,q,1}\) and \(\text{Game}_{\Pi,4,q,2}\) are identically distributed from \(\mathcal{A}\)'s view for any fixed

- \((A, a) \leftarrow D_k,
- \((\{V_0\}) \subseteq \{0, L+2\}, Z) \leftarrow R (Z_p^{(k+1) \times k}) L+3 \times Z_p^{k \times k},
- \text{master secret key } k \leftarrow R Z_p^{k+1},
- \hat{r} \leftarrow R Z_p^k \text{ for creating } s_sk^{(2)}_{\text{ID}_q},
- r_{\text{ID}_q,T} \leftarrow R Z_p^k \text{ for creating } sk_{\text{ID}_q,T} \text{ such that } s_{\text{pa}(\text{ID})} = \text{ID}_q,
- t_{\text{ID}_q,T,\theta}, \tilde{t}_{\text{ID}_q,T,\theta} \leftarrow R Z_p^k \text{ for creating } \text{KU}_{\text{ID}_q,T},
- \alpha \leftarrow R Z_p^k \text{ and } \alpha \leftarrow R Z_p^k.

Specifically, the randomnesses of \(\hat{r}_{\text{ID}_q,T,\theta} \leftarrow R Z_p^k, \text{del}_{\text{ID}_q,T}, \text{delK}_{\text{ID}_q,T} \leftarrow R Z_p^{k+1}\) enable us to prove the claim. We note that all \(sk_{\text{ID}}\) such that \(s_{\text{pa}(\text{ID})} \neq \text{ID}_q\), all \(\text{KU}_{\text{ID}_q,T}\) such that \(\text{ID} \neq \text{ID}_q\), and all \(\text{delK}_{\text{ID}_q,T}\) are created in the same way in both \(\text{Game}_{\Pi,4,q,1}\) and \(\text{Game}_{\Pi,4,q,2}\). We further note that even when \(r_{\text{ID}_q,T} \leftarrow R Z_p^k\) are fixed, \(\text{SK}_{\text{ID},\theta,0}, \text{SK}_{\text{ID},\theta,2}\) do not reveal the quantities of \(\hat{r}_{\text{ID}_q,T} \leftarrow R Z_p^k\) in (10) since they are masked by \(\tilde{r}_{\text{ID}_q,T} \leftarrow R Z_p^k\). Since \(r_{\text{ID}_q,T}\) and \(t_{\text{ID}_q,T,\theta}, \tilde{t}_{\text{ID}_q,T,\theta}, \tilde{t}_{\text{ID}_q,T,\theta}\) are fixed, \(sk_{\text{ID}}\) and \(\text{KU}_{\text{ID}_q,T}\) are distributed in the same way in both \(\text{Game}_{\Pi,4,q,1}\) and \(\text{Game}_{\Pi,4,q,2}\) except \(\text{SK}_{\text{ID},\theta,1}\) and \(\text{KU}_{\text{ID}_q,T,\theta,1}, \text{KU}_{\text{ID}_q,T,\theta,1}\). In \(\text{Game}_{\Pi,4,q,1}\), \(\text{SK}_{\text{ID},\theta,1}\) and \(\text{KU}_{\text{pa}(\text{ID}),T,\theta,1,\text{KU}_{\text{pa}(\text{ID}),T,1}}\) are distributed as follows:

\[
\text{SK}_{\text{ID},\theta,1} = [k_{\text{ID}_q,\theta} + \hat{r}_{\text{ID}_q,\theta} \alpha a]2 \cdot \lfloor [V_0 + \text{id}_{1}V_1 + \cdots + \text{id}_{|\text{ID}|}V_{|\text{ID}|}]Zr_{\text{ID}_q,\theta} \rfloor^2.
\]
where \( \tilde{K}_{ID_i} \), \( \tilde{T}_{ID_i} \), and \( \tilde{r}_{ID_i} \) are distributed in \( \mathbb{Z}_p \) uniformly at random and \( k_{ID_i} + \alpha a^{\perp}, \tilde{K}_{ID_i} - \alpha a^{\perp} \) are distributed in \( \mathbb{Z}_p^{k+1} \) uniformly at random. Therefore, the above distribution is the same as the distribution in Game\(_{II,4,q,2}\) by setting \( \tilde{r}_{ID_i} - 1 \) as the randomnesses in (10) and \( k_{ID_i} + \alpha a^{\perp}, \tilde{K}_{ID_i} - \alpha a^{\perp} \). We note that the claim holds for all ID such that \( \alpha a(\text{ID}) = 0 \) and all nodes \( \theta \in B_{ID_i}, \) simultaneously. Thus, we complete the proof of Lemma 12.

**Lemma 13** (Sub-secret Key Transition from Pseudo-SF to Semi-functional, Game\(_{II,4,q,2} \approx \) Game\(_{II,4,q,3}\)). Game\(_{II,4,q,2}\) and Game\(_{II,4,q,3}\) are computationally indistinguishable under the MDDH assumption in \( \mathbb{G}_2 \). Specifically, for any PPT Type-II adversary \( A \) making at most \( Q_{\text{gen}} \) secret key generation queries, there exists a reduction algorithm \( B_{II,4} \) such that

\[
|\text{Adv}_{II,4,q,2}(\lambda) - \text{Adv}_{II,4,q,3}(\lambda)| \leq \text{Adv}_{B_{II,4}}^{MDDH-G_2}(\lambda) + \frac{2}{p-1}
\]

and \( T(B_{II,4}) \approx T(A) + Q_{\text{gen}}|T| \cdot \text{poly}(\lambda, L) \), where \( \text{poly}(\lambda, L) \) is independent of \( T(A) \).

We omit the detailed proof of Lemma 13 since it is almost the same as the proof of Lemma 11. By combining Lemmata 11–13, we have

\[
|\text{Adv}_{II,4}(\lambda) - \text{Adv}_{II,5}(\lambda)| \\
\leq \sum_{q \in [Q_{\text{gen}}]} |\text{Adv}_{II,4,q-1,3}(\lambda) - \text{Adv}_{II,4,q,1}(\lambda)| + \sum_{q \in [Q_{\text{gen}}]} |\text{Adv}_{II,4,q,1}(\lambda) - \text{Adv}_{II,4,q,2}(\lambda)| \\
+ \sum_{q \in [Q_{\text{gen}}]} |\text{Adv}_{II,4,q,2}(\lambda) - \text{Adv}_{II,4,q,3}(\lambda)| \\
\leq Q_{\text{gen}} \cdot \sum_{j \in [2]} \text{Adv}_{B_{II,j+2}}^{MDDH-G_2}(\lambda) + \frac{4Q_{\text{gen}}}{p-1}.
\]

Thus, we complete the proof of Lemma 10.

**Lemma 14** (Decryption Key Invariance, Game\(_{II,5} \approx \) Game\(_{II,6}\)). Game\(_{II,5}\) and Game\(_{II,6}\) are computationally indistinguishable under the MDDH assumption in \( \mathbb{G}_2 \). Specifically, for any PPT Type-I adversary \( A \) making at most \( Q_{\text{gen}} \) secret key generation queries and \( Q_{\text{dk}} \) decryption key reveal queries, there exists reduction algorithms \( B_{II,5} \) and \( B_{II,6} \) such that

\[
|\text{Adv}_{II,5}(\lambda) - \text{Adv}_{II,6}(\lambda)| \leq Q_{\text{dk}} \cdot \sum_{j \in [2]} \text{Adv}_{B_{II,j+4}}^{MDDH-G_2}(\lambda) + \frac{4Q_{\text{dk}}}{p-1}
\]

and \( \max_{j \in [2]} T(B_{II,j+4}) \approx T(A) + Q_{\text{gen}}|T| \cdot \text{poly}(\lambda, L) \), where \( \text{poly}(\lambda, L) \) is independent of \( T(A) \).
The structure of the proof is the same as the proof of Lemma 5. However, the transition from pseudo-normal to pseudo-SF is a little more complicated since we have to change \( dk_{ID,T} \) for \( ID \in \text{prefix}^+(ID^*) \setminus \{ID^*\} \) to be semi-functional.

Proof of Lemma 14. To prove Lemma 14, we further introduce the following auxiliary distributions.

Pseudo-normal Decryption Keys: A pseudo-normal decryption key \( dk_{ID,T} := (DK_{ID,T,0}, DK_{ID,T,1}, DK_{ID,T,2}, DK'_{ID,T,2}) \) is defined as follows:

\[
\begin{align*}
DK_{ID,T,0} & := [Zu_{ID,T}]_2, \\
DK'_{ID,T,0} & := [Zu'_{ID,T}]_2, \\
DK_{ID,T,1} & := [k]_2 \cdot [(V_0 + id_1 V_1 + \cdots + id_{|ID|} V_{|ID|}) Zu_{ID,T}]_2 \\
& \quad \cdot [(V_0 + TV_{L+1}) Zu_{ID,T}]_2 \cdot [(\hat{u}a)^{-1}]v_0 + v_1 id_1 + \cdots + v_{|ID|} id_{|ID|} + v_{L+1} T, \\
DK_{ID,T,2} & := [V_{L+2} Zu_{ID,T}]_2 \cdot [(\hat{u}a)^{-1}]^{-1},
\end{align*}
\]

where \( u_{ID,T}, u'_{ID,T} \leftarrow_R \mathbb{Z}_p^k \) and \( \hat{u} \leftarrow_R \mathbb{Z}_p^* \). Here, the boxed part denotes the change from the normal decryption key.

Pseudo-SF Decryption Keys: A pseudo-SF decryption key \( dk_{ID,T} := (DK_{ID,T,0}, DK'_{ID,T,0}, DK_{ID,T,1}, DK'_{ID,T,2}) \) is defined as follows:

\[
\begin{align*}
DK_{ID,T,0} & := [Zu_{ID,T}]_2, \\
DK'_{ID,T,0} & := [Zu'_{ID,T}]_2, \\
DK_{ID,T,1} & := [k + (\alpha a)^{-1}]_2 \cdot [(V_0 + id_1 V_1 + \cdots + id_{|ID|} V_{|ID|}) Zu_{ID,T}]_2 \\
& \quad \cdot [(V_0 + TV_{L+1}) Zu'_{ID,T}]_2 \cdot [(\hat{u}a)^{-1}]v_0 + v_1 id_1 + \cdots + v_{|ID|} id_{|ID|} + v_{L+1} T, \\
DK_{ID,T,2} & := [V_{L+2} Zu_{ID,T}]_2 \cdot [(\hat{u}a)^{-1}]^{-1}, \\
DK'_{ID,T,2} & := [V_{L+2} Zu'_{ID,T}]_2 \cdot [(\hat{u}a)^{-1}]^{-1},
\end{align*}
\]

where \( u_{ID,T}, u'_{ID,T} \leftarrow_R \mathbb{Z}_p^k \), \( \hat{u} \leftarrow_R \mathbb{Z}_p^* \), and \( \alpha \leftarrow_R \mathbb{Z}_p^* \). Here, the boxed part denotes the change from the pseudo-normal decryption key.

Let \( (ID_q, T_q) \) denote the tuple on which \( A \) makes the \( q \)-th decryption key reveal query. We further introduce the following sequence of games for \( q \in [Q_{dk}] \):

Game\textsubscript{15,5,1}: This game is the same as Game\textsubscript{15,5} except that
- \( C \) creates semi-functional \( dk_{ID,T} \) upon \( A \)'s first \( q - 1 \) decryption key reveal queries,
- \( C \) creates pseudo-normal \( dk_{ID,T_q} \) upon \( A \)'s \( q \)-th decryption key reveal query,
- \( C \) creates normal \( dk_{ID,T} \) upon \( A \)'s last \( Q_{dk} - q \) decryption key reveal queries.

Game\textsubscript{15,5,2}: This game is the same as Game\textsubscript{15,5,1} except that
- \( C \) creates pseudo-SF \( dk_{ID,T_q} \) upon \( A \)'s \( q \)-th decryption key reveal query.

Game\textsubscript{15,5,3}: This game is the same as Game\textsubscript{15,5,2} except that
- \( C \) creates semi-functional \( dk_{ID,T_q} \) upon \( A \)'s \( q \)-th decryption key reveal query.

We use the notation \( \text{Game\textsubscript{15,5,0,3}} = \text{Game\textsubscript{15,5}} \). By definition, \( \text{Game\textsubscript{15,5,0,3}} = \text{Game\textsubscript{15,6}} \). Hereafter, we prove

\[
\text{Game\textsubscript{15,5,q,1}} \approx c \text{ Game\textsubscript{15,5,q,2}} \equiv \text{Game\textsubscript{15,5,q,2}} \approx c \text{ Game\textsubscript{15,5,q,3}},
\]

where the fact implies that \( \text{Game\textsubscript{15,5}} \approx c \text{ Game\textsubscript{15,6}} \).
Lemma 15 (Decryption Key Transition from Normal to Pseudo-normal, Game\(\text{II}_{5,q}^{1-3} \simeq_c \text{Game}\_\text{II}_{5,q}^{0,1}\)). Game\(\text{II}_{5,q}^{1-3}\) and Game\(\text{II}_{5,q}^{0,1}\) are computationally indistinguishable under the MDDH assumption in \(\mathbb{G}_2\). Specifically, for any PPT Type-II adversary \(A\) making at most \(Q_{\text{gen}}\) secret key generation queries, there exists a reduction algorithm \(B_{\text{II},5}\) such that

\[
|\text{Adv}_{\text{II},5}^{1-3}(\lambda) - \text{Adv}_{\text{II},5}^{0,1}(\lambda)| \leq \text{Adv}_{\text{MDDH}}^{\mathbb{G}_2}(\lambda) + \frac{2}{p - 1}
\]

and \(T(B_{\text{II},5}) \approx T(A) + k^2Q_{\text{gen}}|T| \cdot \text{poly}(\lambda, L)\), where \(\text{poly}(\lambda, L)\) is independent of \(T(A)\).

Proof of Lemma 15. The reduction algorithm \(B_{\text{II},5}\) is given a MDDH instance in \(\mathbb{G}_2\):

\[
(G(1^\lambda), |B|_2, |b|_2 = |Bu + \hat{u}e|_2),
\]

where \(\mathbf{B} \leftarrow_R D_k, \mathbf{u} \leftarrow_R Z_p^{k}, \hat{u} = 0\) or \(\hat{u} \leftarrow_R Z_p, \mathbf{e} = (0, \ldots, 0, 1)^\top \in Z_p^{k+1}\). Hereafter, we assume that \(\hat{u} \leftarrow_R Z_p^*\) in the latter case with the statistical difference \(1/p\).

\(B_{\text{II},5}\) creates MPK and \(ct^*\) in the same way as the proof of Lemma 6. \(B_{\text{II},5}\) creates \(sk_{\text{ID}, \tau, 0}\) by computing (5) and \(sk_{\text{ID}, \tau, 0}\) such that \(|\text{ID}| \geq 1\) in the same way as the real scheme. \(B_{\text{II},5}\) creates \(sk_{\text{ID}, \tau, 1}\) by computing (9) and computes \(sk_{\text{ID}, \tau} = (DK_{\text{ID}, \tau, 0}, DK'_{\text{ID}, \tau, 0}, DK_{\text{ID}, \tau, 1}, DK_{\text{ID}, \tau, 2}, DK'_{\text{ID}, \tau, 2})\):

\[
\begin{align*}
DK_{\text{ID}, \tau, 0} &= |B|_2 \cdot |Z\hat{u}_{\text{ID}, \tau}|_2, & DK'_{\text{ID}, \tau, 0} &= |B|_2 \cdot |Z\hat{u}'_{\text{ID}, \tau}|_2, \\
DK_{\text{ID}, \tau, 1} &= |k|_2 \cdot [(\mathbf{V}_0 + \mathbf{u}_q, 1)\mathbf{V}_1 + \cdots + \mathbf{u}_q|\text{ID}| \mathbf{V}_{|\text{ID}|}]B_2 \\
&\quad \cdot [(V_0 + \mathbf{u}_q, 1)\mathbf{V}_1 + \cdots + \mathbf{u}_q|\text{ID}| \mathbf{V}_{|\text{ID}|}]Z\hat{u}_{\text{ID}, \tau} |_2, \\
DK_{\text{ID}, \tau, 2} &= [V_{L+2}B_2 \cdot [-a^+ b]_2 \cdot [V_{L+2}Z\hat{u}_{\text{ID}, \tau}] |_2, \\
DK'_{\text{ID}, \tau, 2} &= [V_{L+2}B_2 \cdot [-a^+ b]_2 \cdot [V_{L+2}Z\hat{u}'_{\text{ID}, \tau}] |_2.
\end{align*}
\]

By following the same argument in the proof of Lemma 6, \(dk_{\text{ID}, \tau, q}\) is a normal decryption key as in Game\(\text{II}_{5,q}^{1-3}\) if \(\hat{u} = 0\), and pseudo-normal decryption key as in Game\(\text{II}_{5,q}^{0,1}\) if \(\hat{u} \leftarrow_R Z_p^*\), by setting \(\hat{u}_{\text{ID}, \tau} = \hat{Z}^{-1}u + \hat{u}_{\text{ID}, \tau}\) and \(\hat{u}'_{\text{ID}, \tau} = \hat{Z}^{-1}u + \hat{u}'_{\text{ID}, \tau}\).

- If \(m < q\), \(B_{\text{II},5}\) creates normal \(dk_{\text{ID}, m, \tau, m}\) by computing (13).

Thus, we complete the proof of Lemma 15. \(\square\)

Lemma 16 (Decryption Key Transition from Pseudo-normal to Pseudo-SF, Game\(\text{II}_{5,q}^{0,1} \equiv \text{Game}\_\text{II}_{5,q}^{0,2}\)). Game\(\text{II}_{5,q}^{0,1}\) and Game\(\text{II}_{5,q}^{0,2}\) are identically distributed from \(A\)'s view. Specifically, for any Type-II adversary \(A\), it holds that

\[
\text{Adv}_{\text{II},5,q,1}(\lambda) = \text{Adv}_{\text{II},5,q,2}(\lambda).
\]

37
Proof of Lemma 16. Here, we prove a stronger claim that \( \text{Game}_{\Pi,5,q,1} \) and \( \text{Game}_{\Pi,5,q,2} \) are identically distributed for any fixed

- \((A, a) \leftarrow_R D_k,
- ((V_t)_{t \in [0, L+2]}, Z) \leftarrow_R (Z_p^{(k+1) \times k})^{L+3} \times Z_p^k,
- \) master secret key \( k \leftarrow_R Z_p^{k+1},
- \) \( \hat{u} \leftarrow_R Z_p^* \) for creating \( dk_{\text{id}_q, \tau_q},
- \) \( c \leftarrow_R Z_p^{k+1} \) for creating the challenge ciphertext,
- \( (\text{id}^*, T^*, M^*_0, M^*_1) \in Z_p^2 \times M^2 \) and random coin \( \text{coin} \leftarrow_R \{0, 1\},
- \) \( u_{\text{id}_q, \tau_q}, u'_{\text{id}_q, \tau_q} \leftarrow_R Z_p^k \) and \( \hat{u} \leftarrow_R Z_p^* \) for creating \( q \)-th queried \( dk_{\text{id}_q, \tau_q} \).

Specifically, the randomness of \((v_0, v_1, \ldots, v_{L+1}) \leftarrow_R Z_p^{L+2}\) enables us to prove the claim. Since all \( \text{sk}_{\text{id}}, \text{ku}_{\text{id}, T}, \) and \( dk_{\text{id}, T} \) except \( dk_{\text{id}_q, \tau_q} \) are created in the same way in both \( \text{Game}_{\Pi,5,q,1} \) and \( \text{Game}_{\Pi,5,q,2} \), and all the other elements have been already fixed, it is sufficient to show that

\[
\begin{cases}
 v_0 + \text{id}_{q,1} v_1 + \cdots + \text{id}_{\text{id}^*, v_{q,1} v_{L+1}}, v_0 + T^* v_{L+1}, \\
 v_0 + \text{id}_{q,1} v_1 + \cdots + \text{id}_{\text{id}^*, v_{q,1} v_{L+1}} + T_q v_{L+1}
\end{cases}
\]

where \((v_0, v_1, \ldots, v_{L+1}) \leftarrow_R Z_{p}^{L+2}\). Here, the first two elements are \( \text{tag}, \text{tag}' \) and last element is the exponent of \( |\text{ia}^*|_2 \) of \( DK_{\text{id}_q, \tau_q, 1} \) in \( \text{Game}_{\Pi,5,q,1} \) and \( \text{Game}_{\Pi,5,q,2} \), respectively. If \( \text{id}_q \notin \text{prefix}^+(\text{id}^*) \) holds, \( \{v_0 + \text{id}_{q,1} v_1 + \cdots + \text{id}_{\text{id}^*, v_{q,1} v_{L+1}}, v_0 + \text{id}_{q,1} v_1 + \cdots + \text{id}_{\text{id}^*, v_{q,1} v_{L+1}} + T_q v_{L+1}\}\) is distributed in \( Z_p^2 \) uniformly at random by following the standard argument for proving HIBE. If \( T_q \neq T^* \) holds, \( \{v_0 + T^* v_{L+1}, v_0 + T_q v_{L+1}\}\) is distributed in \( Z_p^2 \) uniformly at random by following the standard argument for proving IBE. If \( \text{id}_q \in \text{prefix}^+(\text{id}^*) \setminus \{\text{id}^*\}, \{v_0 + \text{id}_{q,1} v_1 + \cdots + \text{id}_{\text{id}^*, v_{q,1} v_{L+1}}, v_0 + \text{id}_{q,1} v_1 + \cdots + \text{id}_{\text{id}^*, v_{q,1} v_{L+1}} + T_q v_{L+1}\}\) is distributed in \( Z_p^2 \) uniformly at random due to the random \( \text{id}_{q,1} v_{q,1} v_{L+1} + \text{id}_{\text{id}^*, v_{q,1} v_{L+1}} \). Since \( (\text{id}_q, T_q) \neq (\text{id}^*, T^*) \) holds due to the security definition of RHIBE, we have proved the claim. Thus, we complete the proof of Lemma 16.

Lemma 17 (Decryption Key Transition from Pseudo-SF to Semi-functional, \( \text{Game}_{\Pi,5,q,2} \approx_2 \text{Game}_{\Pi,5,q,3} \)). \( \text{Game}_{\Pi,5,q,2} \) and \( \text{Game}_{\Pi,5,q,3} \) are computationally indistinguishable under the MDDH assumption in \( G_2 \). Specifically, for any PPT Type-II adversary \( A \) making at most \( Q_{\text{gen}} \) secret key generation queries, there exists a reduction algorithm \( B_{\Pi,6} \) such that

\[
|\text{Adv}_{\Pi,5,q,2}(\lambda) - \text{Adv}_{\Pi,5,q,3}(\lambda)| \leq \text{Adv}_{\Pi,6}^{\text{MDDH-G}2}(\lambda) + \frac{2}{p - 1}
\]

and \( T(B_{\Pi,6}) \approx T(A) + k^2 Q_{\text{gen}} |T| \cdot \text{poly}(\lambda, L) \), where \( \text{poly}(\lambda, L) \) is independent of \( T(A) \).

We omit the detailed proof of Lemma 17 since it is almost the same as the proof of Lemma 15. The only difference is that \( B_{\Pi,6} \) creates \( dk_{\text{id}_q, \tau_q} \) upon \( A \)'s \( q \)-th decryption key reveal query by

\[
\begin{align*}
\text{DK}_{\text{id}_q, \tau_q, 1} &= [k + \text{oa}^*]_2 \cdot [(V_0 + \text{id}_{q,1} V_1 + \cdots + \text{id}_{\text{id}_q} V_{\text{id}_q}]_2 \cdot [V_0 + T_q V_{L+1}]_2 \cdot [a^*]_2 \\
&\cdot [V_0 + \text{id}_{q,1} V_1 + \cdots + \text{id}_{\text{id}_q} V_{\text{id}_q]_2 \\
&\cdot [(V_0 + T_q V_{L+1}]_2 \cdot \text{poly}(\lambda, L)
\end{align*}
\]
where the boxed parts denote the changes from (17). If \( \hat{q} \leftarrow R \mathbb{Z}_p^* \), \( dk_{1D,q} \) is a pseudo-SF decryption key as in \( \text{Game}_{II,5,q,2} \). If \( q = 0 \), \( dk_{1D,q} \) is a semi-functional decryption key as in \( \text{Game}_{II,5,q,3} \).

By combining Lemmata 15–17, we have

\[
|\text{Adv}_{II,5}(\lambda) - \text{Adv}_{II,6}(\lambda)| \\
\leq \sum_{q \in [Q_{bk}]} |\text{Adv}_{II,5,q-1,3}(\lambda) - \text{Adv}_{II,5,q-1}(\lambda)| + \sum_{q \in [Q_{bk}]} |\text{Adv}_{II,5,q,1}(\lambda) - \text{Adv}_{II,5,q,2}(\lambda)| \\
+ \sum_{q \in [Q_{bk}]} |\text{Adv}_{II,5,q,2}(\lambda) - \text{Adv}_{II,5,q,3}(\lambda)| \\
\leq Q_{dk} \cdot \sum_{j \in [2]} \text{Adv}^{MDDH-G_2}_{II,j+4}(\lambda) + \frac{4Q_{dk}}{p - 1}.
\]

Thus, we complete the proof of Lemma 14.

**Lemma 18** (Final Transition, \( \text{Game}_{II,6} \equiv \text{Game}_{II,7} \)). \( \text{Game}_{II,6} \) and \( \text{Game}_{II,7} \) are identically distributed. Specifically, for any Type-II adversary \( \mathcal{A} \), it holds that

\[\text{Adv}_{II,6}(\lambda) = \text{Adv}_{II,7}(\lambda).\]

**Proof of Lemma 18.** Run \( (p, G_1, G_2, G_T, g_1, g_2, e) \leftarrow G(1^\lambda) \) and sample \( (A, a^\perp) \leftarrow D_k \), \((\langle V \rangle_{\ell \in [0, L + 2]}, Z) \leftarrow R (Z_p^{k+1})^{L+3} \times Z_p^{k \times k}, k \leftarrow R Z_p^{k+1}, \) and \( \alpha \leftarrow R Z_p^* \). We set \( \text{MSK} = k - \alpha a^\perp \) and returns

\[\text{MPK} = \left( [A], ([V^T A]_{\ell \in [0, L + 2]}, [Z]_2, ([V, Z]_2)_{\ell \in [0, L + 2]}, [A^T k]_T \right)\]

to \( \mathcal{A} \). Since it holds that

\[\left[a^\perp k\right]_T = e([A]_1, [k]_2) = e([A]_1, [k]_2) \cdot e([A]_1, [a^\perp]_2^\perp) = e([A]_1, [k - \alpha a^\perp]_2) = [A^T \text{MSK}]_T\]

MPK follows the same distribution as the real scheme. Furthermore, MPK does not reveal the quantity of \( \alpha \) in both \( \text{Game}_{II,6} \) and \( \text{Game}_{II,7} \). We create \( s, sk_{1D}^{(1)} \) by computing (9) and create \( sk_{1D,0} \) by computing (10). We create \( ku_{1D,0} \) such that \( [ID] \geq 1 \) in the same way as the real scheme. In \( \text{Game}_{II,6} \), \( C \) uses \( \text{MSK} \) only for computing semi-functional \( ku_{k_{BC,T,0}} \) (5), semi-functional \( ku_{1D,T} \) (6), and semi-functional \( dk_{1D,T} \) (7). In this proof, since \([k]_2 = \text{MSK} + \alpha a^\perp\), we use \([k]_2\) and create \( ku_{k_{BC,T,0}}, ku_{1D,T} \), and \( dk_{1D,T} \) by computing (11), (12), and (13), and they follow semi-functional distribution as in (5), (6), and (7), respectively.

Summarizing the creations so far, we do not use \( \text{MSK} \) for creating all \( \text{MPK} \), \( sk_{1D,0} \), \( ku_{k_{BC,T,0}} \), \( ku_{1D,T,0} \), \( ku_{1D,T} \), and \( dk_{1D,T} \). In other words, the quantity of \( \alpha \) is not revealed to \( \mathcal{A} \) so far. In both \( \text{Game}_{II,6} \) and \( \text{Game}_{II,7} \), \( ct^* \) follows the same distribution except \( C_2 \). In \( \text{Game}_{II,6} \), \( C_2 \) is distributed as follows:

\[C_2 = M_{coin} \cdot [c^T \text{MSK}]_T = \left( M_{coin} \cdot [-\alpha c^T a^\perp]_T \right) \cdot [c^T k]_T.\]

Since \( \alpha \in Z_p^* \), \( -\alpha a^\perp = 0 \) holds only when \( c^T a^\perp = 0 \). Since \( c \) is distributed in \( Z_p^{k+1} \) uniformly at random, it holds that \( c^T a^\perp = 0 \) with probability \( 1/p \). In contrast, when \( c^T a^\perp \neq 0 \), \( -\alpha a^\perp \) for \( \alpha \leftarrow R Z_p^* \) is distributed in \( Z_p^* \) uniformly at random. Then, \( -\alpha a^\perp \) becomes each non-zero value
with probability \( (1 - \frac{1}{p}) \cdot \frac{1}{p-1} = \frac{1}{p} \). Therefore, \( M_{\text{coin}} \cdot [-\alpha \mathbf{c}^T \mathbf{a}]_T \) is distributed in \( G_T \) uniformly at random. Thus, we complete the proof of Lemma 18. 

By combining with Lemmata 3, 4, 5, 9, 10, 14, and 18, we have

\[
\text{Adv}^{\text{RHIBE}}_{\Pi, L, A}(\lambda) \\
\leq \sum_{i \in [7]} |\text{Adv}_{\Pi, i-1}(\lambda) - \text{Adv}_{\Pi, i}(\lambda)| + \text{Adv}_{\Pi, 7}(\lambda)
\]
\[
\leq \text{Adv}_{\Pi, 0}^{\text{MDDH-G}_1}(\lambda) + Q_{\text{rev}} \cdot \sum_{j \in [2]} \text{Adv}_{\Pi, j}^{\text{MDDH-G}_2}(\lambda) + Q_{\text{gen}} \cdot \sum_{j \in [2]} \text{Adv}_{\Pi, j+2}^{\text{MDDH-G}_2}(\lambda)
\]
\[
+ Q_{\text{dk}} \cdot \sum_{j \in [2]} \text{Adv}_{\Pi, j+4}^{\text{MDDH-G}_2}(\lambda) + \frac{4(Q_{\text{rev}} + Q_{\text{gen}} + Q_{\text{dk}})}{p}.
\]

By definition, \( Q_{\text{rev}} \leq Q_{\text{gen}} \) and \( Q_{\text{rev}} \leq Q_{\text{gen}} |T| \) hold. Therefore, it holds that

\[
\text{Adv}^{\text{RHIBE}}_{\Pi, L, A}(\lambda)
\leq \text{Adv}_{\Pi, 0}^{\text{MDDH-G}_1}(\lambda) + Q_{\text{gen}} \left( \sum_{j \in [4]} \text{Adv}_{\Pi, j}^{\text{MDDH-G}_2}(\lambda) + |T| \cdot \sum_{j \in [2]} \text{Adv}_{\Pi, j+4}^{\text{MDDH-G}_2}(\lambda) \right)
\]
\[
+ O\left( \frac{Q_{\text{gen}} |T|}{p} \right).
\]

Thus, we complete the proof of Lemma 2.

\[\square\]

6 Adaptive Security against the Type-I Adversary

We reposit the definition of the Type-I adversary:

Type-I Adversary: \( \mathcal{A} \) is called Type-I if it makes the secret key reveal queries on some \( \text{ID} \in \text{prefix}^+ (\text{ID}^*) \).

It is clear that our proof strategy against the Type-II adversary is insufficient for proving the adaptive security against the Type-I adversary since \( \mathcal{A} \) receives \( \hat{\text{sk}}_{\text{ID}} \) such that \( \text{ID} \in \text{prefix}^+ (\text{ID}^*) \). Although we do not perform a detailed analysis of this problem, we believe that by combining the proof technique of Emura et al. [ETW20] against the Type-I adversary and the semi-functional randomness switching, we may be able to prove the adaptive security of our RHIBE scheme against the Type-I adversary. As we claimed in Remark 2, Emura et al. divided the Type-I adversary into the Type-I-\( \ell^* \) adversary for \( \ell^* \in [L] \) such that \( \mathcal{A} \) makes the secret key reveal a query on \( \text{ID}^*_{[\ell^*-1]} \), while \( \mathcal{A} \) does not make the secret key reveal queries on any \( \text{ID}^*_{[\ell]} \) for \( \ell \in [\ell^* - 1] \). Thus, Emura et al.’s proof technique inherently suffers from \( O(L) \) reduction loss. This reduction loss is unavoidable for their proof technique since they used the value \( \ell^* \) to define the way in which the reduction algorithm answers \( \mathcal{A} \)’s key queries.

We adopted another approach for achieving tighter reduction. Let \( \text{ID}_q \) denote the identity on which \( \mathcal{A} \) makes \( q \)-th secret key generation query. First, we determine the number \( Q^* \in [Q_{\text{gen}}] \) such that \( \text{ID}_{Q^*} = \text{ID}^*_{[\ell^*-1]} \) with \( Q_{\text{gen}} \) reduction loss. Although we also use the value \( \ell^* \) to design the manner in which the reduction algorithm answers \( \mathcal{A} \)’s key queries, as done by Emura et al., our proof does not suffer from \( O(L) \) reduction loss since the reduction algorithm answers all the key queries of \( \mathcal{A} \) in the same manner until \( \mathcal{A} \)’s \( Q^* \)-th secret key generation query. By definition of \( \text{ID}^*_{[\ell^*-1]} \),
all ID on which \( A \) makes the secret key reveal queries satisfy \( ID \notin prefix^+(ID^*) \) until \( A \)'s \( Q^* \)-th secret key generation query. After \( A \)'s \( Q^* \)-th secret key generation query, we can detect whether \( ID^*_{[\ell^*]} \notin prefix^+(ID) \) holds for any ID. Thus, upon \( A \)'s secret key reveal queries on ID, we change \( sk_{ID} \) to be semi-functional only when it holds that \( ID^*_{[\ell^*]} \notin prefix^+(ID) \).

Next, we explain how to change all \( ku_{ID,T} \) to be semi-functional. In this case, our proof technique against the Type-II adversary is still insufficient for proving adaptive security against the Type-I adversary. By definition of \( ID^*_{[\ell^*]} \), \( sk_{ID^*_{[\ell^*]-1}} \) that includes the delegation keys \( del_{ID^*_{[\ell^*]-1}} \theta \) is not revealed to \( A \). Nevertheless, we cannot apply the semi-functional randomness switching to change \( ku_{ID^*_{[\ell^*]-1};T} \) to be semi-functional since we cannot change \( sk_{ID^*_{[\ell^*]}} \) to be semi-functional. To overcome this problem, we use the information derived from the guess of \( Q^* \). After \( A \)'s \( Q^* \)-th secret key generation query, we can detect the time period \( T_{RL} \) when \( ID^*_{[\ell^*]} \) is revoked. From the security definition of RHIBE, since \( ID^*_{[\ell^*]} \) has to be revoked by the challenge time period \( T^* \), \( T_{RL} \leq T^* \) holds. In other words, we can use the fact \( T \neq T^* \) to change all \( ku_{ID,T} \) and \( dk_{ID,T} \) to be semi-functional before the time period \( T_{RL} \). From the above discussion, \( A \) receives normal \( sk_{ID^*_{[\ell^*]}} \). On the other hand, after time period \( T_{RL} \), all \( ku_{ID^*_{[\ell^*]-1};T,\theta} \) and \( sk_{ID^*_{[\ell^*]};\theta} \) do not share the same node since \( ID^*_{[\ell^*]} \) is already revoked. Based on this fact, we can apply semi-functional randomness switching to change all \( ku_{ID^*_{[\ell^*]-1};T} \) to be semi-functional.

Following this argument, we prove the adaptive security of our RHIBE scheme. Before providing an overview of our proof, we introduce the following seed key update and its semi-functional distribution.

**Normal Seed Key Updates:** A normal seed key update is defined as \( s.ku_T := (s.KU_{T,0}, s.KU_{KU,1}, s.KU_{KU,2}) \):

\[
\begin{align*}
  s.KU_{T,0} & := [Z_{T}]_2, \\
  s.KU_{T,1} & := [(V_0 + TV_{L+1})Z_{T}]_2, \\
  s.KU_{T,2} & := [V_{L+2}Z_{T}]_2,
\end{align*}
\]

where \( t_T \leftarrow_R \mathbb{Z}_p^k \).

**Semi-functional Seed Key Updates:** A semi-functional seed key update is defined as \( s.ku_T := (s.KU_{T,0}, s.KU_{KU,1}, s.KU_{KU,2}) \):

\[
\begin{align*}
  s.KU_{T,0} & := [Z_{T}]_2, \\
  s.KU_{T,1} & := [\alpha a^+]_2 - [(V_0 + TV_{L+1})Z_{T}]_2, \\
  s.KU_{T,2} & := [V_{L+2}Z_{T}]_2,
\end{align*}
\]

where \( t_T \leftarrow_R \mathbb{Z}_p^k \) and \( \alpha \leftarrow_R \mathbb{Z}_p^\ast \). Here, the term in the box denotes the change from the normal seed key update.

We use the following sequence of games to prove the adaptive security against the Type-I adversary:

- **Game1**\(_{0}\): This is the real security game between the challenger \( C \) and adversary \( A \).
- **Game1**\(_{1}\): This game is the same as **Game1**\(_{0}\) except that the challenge ciphertext \( ct^* \) is semi-functional.
- **Game1**\(_{2}\): Let \( ID^*_{[\ell^*]} \in prefix^+(ID^*) \) denote an identity such that \( A \) makes the secret key reveal queries on \( ID^*_{[\ell^*]} \), while \( A \) does not make the secret key reveal any query on any \((ID^*_{[\ell^*]}), (t^*_{[\ell^*]-1}) \). Let \( ID_q \) denote the identity on which \( A \) makes \( q \)-th secret key generation query. This game is the same as **Game1**\(_{1}\) except that \( C \) guesses the number \( Q^* \) such that \( ID_{Q^*} = ID^*_{[\ell^*]} \). If the guess is not correct, \( C \) aborts the game and outputs a random bit \( \widehat{coin} \leftarrow_R \{0, 1\} \). Hereafter, let
**T_{RL}** denote the first time period such that ID_{Q*} \in RL_{T_{RL}}. From the definition of the Type-I adversary, it holds that T_{RL} \leq T^* if the guess is correct. Hereafter, we describe the case only when the guess is correct.

**Game_{1,3}:** This game is the same as Game_{1,2} except that C modifies the method of creating secret keys sk_{1D}, key updates ku_{1D,T}, and decryption keys dk_{1D,T} as follows:

---

**Table 4:** Distributions of ct*, s.sk_{1D}^{(1)} for creating sk_{1D,θ}, and sk_{1D,θ} in each game in the proof against the Type-I adversary. In the column ct*, we specify the distribution and encrypted plaintext. In the other columns, we specify the distributions and semi-functional randomness of s.sk_{1D} and sk_{1D,θ}.

<table>
<thead>
<tr>
<th>Game</th>
<th>ct*</th>
<th>s.sk_{1D}^{(1)} for ID_{T^*}≠ prefix^{+}(ID)</th>
<th>sk_{1D,θ} for ID_{T^*}≠ prefix^{+}(ID)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Game_{1,0}</td>
<td>normal</td>
<td>normal</td>
<td>normal</td>
</tr>
<tr>
<td></td>
<td>M_{coin}^*</td>
<td>normal</td>
<td>normal</td>
</tr>
<tr>
<td>Game_{1,1}</td>
<td>semi-functional</td>
<td>normal</td>
<td>normal</td>
</tr>
<tr>
<td></td>
<td>M_{coin}^*</td>
<td>normal</td>
<td>normal</td>
</tr>
<tr>
<td>Game_{1,2}</td>
<td>semi-functional</td>
<td>normal</td>
<td>normal</td>
</tr>
<tr>
<td></td>
<td>M_{coin}^*</td>
<td>normal</td>
<td>normal</td>
</tr>
<tr>
<td>Game_{1,3}</td>
<td>semi-functional</td>
<td>semi-functional</td>
<td>semi-functional</td>
</tr>
<tr>
<td></td>
<td>M_{coin}^*</td>
<td>(\alpha \leftarrow R Z_p^*)</td>
<td>(\tilde{r}_{ID,θ} \alpha \leftarrow R Z_p)</td>
</tr>
<tr>
<td>Game_{1,4}</td>
<td>semi-functional</td>
<td>semi-functional</td>
<td>semi-functional</td>
</tr>
<tr>
<td></td>
<td>M_{coin}^*</td>
<td>(\alpha \leftarrow R Z_p^*)</td>
<td>(\tilde{r}_{ID,θ} \alpha \leftarrow R Z_p)</td>
</tr>
<tr>
<td>Game_{1,5}</td>
<td>semi-functional</td>
<td>semi-functional</td>
<td>semi-functional</td>
</tr>
<tr>
<td></td>
<td>M_{coin}^*</td>
<td>(\alpha \leftarrow R Z_p^*)</td>
<td>(\tilde{r}_{ID,θ} \alpha \leftarrow R Z_p)</td>
</tr>
<tr>
<td>Game_{1,6}</td>
<td>semi-functional</td>
<td>semi-functional</td>
<td>semi-functional</td>
</tr>
<tr>
<td></td>
<td>M_{coin}^*</td>
<td>(\alpha \leftarrow R Z_p^*)</td>
<td>(\tilde{r}_{ID,θ} \alpha \leftarrow R Z_p)</td>
</tr>
<tr>
<td>Game_{1,7}</td>
<td>semi-functional</td>
<td>semi-functional</td>
<td>semi-functional</td>
</tr>
<tr>
<td></td>
<td>M_{coin}^*</td>
<td>(\alpha \leftarrow R Z_p^*)</td>
<td>(\tilde{r}_{ID,θ} \alpha \leftarrow R Z_p)</td>
</tr>
<tr>
<td>Game_{1,8}</td>
<td>semi-functional</td>
<td>semi-functional</td>
<td>semi-functional</td>
</tr>
<tr>
<td></td>
<td>M_{coin}^*</td>
<td>(\alpha \leftarrow R G_T)</td>
<td>(\tilde{r}_{ID,θ} \alpha \leftarrow R Z_p)</td>
</tr>
</tbody>
</table>

**Secret Key Creation:** Upon A’s secret key *generation* queries on ID, C does not create sub-secret keys sk_{1D,θ}. Upon A’s secret key *reveal* queries on ID, C first creates normal seed secret keys s.sk_{1D}^{(1)}. Then, C uses s.sk_{1D}^{(1)} to create all sub-secret keys sk_{1D,θ}.

**Key Update and Decryption Key Creation:** C proceeds as follows.

- For each time period T < T_{RL} upon the setup and A’s revoke & key update queries, C first creates normal seed key updates s.ku_{T}. To create ku_{1D,T} (including ku_{k^c,T}) for T < T_{RL}, C uses the seed key updates s.ku_{T} for computing the sub-key updates ku_{1D,T,θ}. It also creates the helper key updates ku_{1D,T} such that |ID| ≥ 1 in the same way as in
the real scheme. To create $\text{dk}_{\text{ID},T}$ for $T < T_{\text{RL}}$, $C$ uses the seed key updates $s.ku_{T}$ for computing the decryption keys $\text{dk}_{\text{ID},T}$.

- Upon $A$’s secret key generation queries on $\text{ID}$, $C$ first creates normal seed keys $s.sk_{\text{ID}}^{(2)}$. $C$ creates $ku_{\text{kgc},T}$ for $T \geq T_{\text{RL}}$ in the same manner as in the real scheme. To create $ku_{\text{ID},T}$ such that $|\text{ID}| \geq 1$ and $\text{dk}_{\text{ID},T}$ for all $T \geq T_{\text{RL}}$, $C$ uses the seed secret keys $s.sk_{\text{ID}}^{(2)}$ for computing the helper decryption keys $\overline{\text{dk}}_{\text{ID},T}$ and creating the sub-key updates $ku_{\text{ID},T,\theta}$ in the same way as in the real scheme.

Game$_{1,4}$: This game is the same as Game$_{1,3}$ except that $C$ creates semi-functional seed secret keys $s.sk_{\text{ID}}^{(1)}$ upon $A$’s secret key reveal queries if $\text{ID}_{Q} \not\in \text{prefix}^{+}(\text{ID})$ holds.

Game$_{1,5}$: This game is the same as Game$_{1,4}$ except that $C$ creates semi-functional seed key updates $s.ku_{T}$. Furthermore, each helper key update $\overline{ku}_{\text{ID},T}$ for $T < T_{\text{RL}}$ is also semi-functional.

Game$_{1,6}$: This game is the same as Game$_{1,5}$ except that $C$ creates semi-functional $ku_{\text{kgc},T}$ for $T \geq T_{\text{RL}}$.

Game$_{1,7}$: This game is the same as Game$_{1,6}$ except that $C$ creates semi-functional helper key updates $\overline{ku}_{\text{ID},T}$ and decryption keys $\text{dk}_{\text{ID},T}$ for $T \geq T_{\text{RL}}$. 

43
Table 5: Distributions of $ku_{ID,T,\theta}$, $\overline{ku}_{ID,T}$ for $|ID| \geq 1$, and $dk_{ID,T}$ for $T < T_{RL}$ in each game in the proof against the Type-I adversary. We specify the distributions and semi-functional randomness of $ku_{kgc,T,\theta}$, $\overline{ku}_{ID,T}$, and $dk_{ID,T}$.

<table>
<thead>
<tr>
<th>Game</th>
<th>$ku_{ID,T,\theta}$ for $T &lt; T_{RL}$</th>
<th>$\overline{ku}<em>{ID,T}$ for $T &lt; T</em>{RL}$</th>
<th>$dk_{ID,T}$ for $T &lt; T_{RL}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Game$_{1,0}$</td>
<td>normal</td>
<td>normal</td>
<td>normal</td>
</tr>
<tr>
<td>Game$_{1,1}$</td>
<td>normal</td>
<td>normal</td>
<td>normal</td>
</tr>
<tr>
<td>Game$_{1,2}$</td>
<td>normal</td>
<td>normal</td>
<td>normal</td>
</tr>
<tr>
<td>Game$_{1,3}$</td>
<td>normal</td>
<td>normal</td>
<td>normal</td>
</tr>
<tr>
<td>Game$_{1,4}$</td>
<td>normal</td>
<td>normal</td>
<td>normal</td>
</tr>
<tr>
<td>Game$_{1,5}$</td>
<td>semi-functional</td>
<td>semi-functional</td>
<td>semi-functional</td>
</tr>
<tr>
<td></td>
<td>$\tilde{t}_{ID,T,\theta} \alpha$</td>
<td>$\tilde{t}_{ID,T} \alpha$</td>
<td>$\tilde{u}_{ID,T} \alpha$</td>
</tr>
<tr>
<td></td>
<td>$\tilde{t}_{ID,T,\theta} \leftarrow R \mathbb{Z}_p$</td>
<td>$\tilde{t}_{ID,T} \leftarrow R \mathbb{Z}_p$</td>
<td>$\tilde{u}_{ID,T} \leftarrow R \mathbb{Z}_p$</td>
</tr>
<tr>
<td>Game$_{1,6}$</td>
<td>semi-functional</td>
<td>semi-functional</td>
<td>semi-functional</td>
</tr>
<tr>
<td></td>
<td>$\tilde{t}_{ID,T,\theta} \alpha$</td>
<td>$\tilde{t}_{ID,T} \alpha$</td>
<td>$\tilde{u}_{ID,T} \alpha$</td>
</tr>
<tr>
<td></td>
<td>$\tilde{t}_{ID,T,\theta} \leftarrow R \mathbb{Z}_p$</td>
<td>$\tilde{t}_{ID,T} \leftarrow R \mathbb{Z}_p$</td>
<td>$\tilde{u}_{ID,T} \leftarrow R \mathbb{Z}_p$</td>
</tr>
<tr>
<td>Game$_{1,7}$</td>
<td>semi-functional</td>
<td>semi-functional</td>
<td>semi-functional</td>
</tr>
<tr>
<td></td>
<td>$\tilde{t}_{ID,T,\theta} \alpha$</td>
<td>$\tilde{t}_{ID,T} \alpha$</td>
<td>$\tilde{u}_{ID,T} \alpha$</td>
</tr>
<tr>
<td></td>
<td>$\tilde{t}_{ID,T,\theta} \leftarrow R \mathbb{Z}_p$</td>
<td>$\tilde{t}_{ID,T} \leftarrow R \mathbb{Z}_p$</td>
<td>$\tilde{u}_{ID,T} \leftarrow R \mathbb{Z}_p$</td>
</tr>
<tr>
<td>Game$_{1,8}$</td>
<td>semi-functional</td>
<td>semi-functional</td>
<td>semi-functional</td>
</tr>
<tr>
<td></td>
<td>$\tilde{t}_{ID,T,\theta} \alpha$</td>
<td>$\tilde{t}_{ID,T} \alpha$</td>
<td>$\tilde{u}_{ID,T} \alpha$</td>
</tr>
<tr>
<td></td>
<td>$\tilde{t}_{ID,T,\theta} \leftarrow R \mathbb{Z}_p$</td>
<td>$\tilde{t}_{ID,T} \leftarrow R \mathbb{Z}_p$</td>
<td>$\tilde{u}_{ID,T} \leftarrow R \mathbb{Z}_p$</td>
</tr>
</tbody>
</table>
Table 6: Distributions of $k_{ku,T,\theta}$, $ku_{ID,T}$ for $|ID| \geq 1$, and $dk_{ID,T}$ for $T \geq T_{RL}$ in each game in the proof against the Type-I adversary. In the columns, we specify the distributions and semi-functional randomness of $k_{ku,T,\theta}$, $ku_{ID,T}$, and $dk_{ID,T}$.

<table>
<thead>
<tr>
<th>Game</th>
<th>$k_{ku,T,\theta}$ for $T \geq T_{RL}$</th>
<th>$ku_{ID,T}$ for $T \geq T_{RL}$</th>
<th>$dk_{ID,T}$ for $T \geq T_{RL}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Game_{1,0}</td>
<td>normal</td>
<td>normal</td>
<td>normal</td>
</tr>
<tr>
<td>Game_{1,1}</td>
<td>normal</td>
<td>normal</td>
<td>normal</td>
</tr>
<tr>
<td>Game_{1,2}</td>
<td>normal</td>
<td>normal</td>
<td>normal</td>
</tr>
<tr>
<td>Game_{1,3}</td>
<td>normal</td>
<td>normal</td>
<td>normal</td>
</tr>
<tr>
<td>Game_{1,4}</td>
<td>normal</td>
<td>normal</td>
<td>normal</td>
</tr>
<tr>
<td>Game_{1,5}</td>
<td>normal</td>
<td>normal</td>
<td>normal</td>
</tr>
<tr>
<td>Game_{1,6}</td>
<td>semi-functional</td>
<td>normal</td>
<td>normal</td>
</tr>
<tr>
<td></td>
<td>$\alpha \leftarrow R \mathbb{Z}_p^*$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Game_{1,7}</td>
<td>semi-functional</td>
<td>semi-functional</td>
<td>semi-functional</td>
</tr>
<tr>
<td></td>
<td>$\alpha \leftarrow R \mathbb{Z}_p^*$</td>
<td>$\alpha \leftarrow R \mathbb{Z}_p^*$</td>
<td>$\alpha \leftarrow R \mathbb{Z}_p^*$</td>
</tr>
<tr>
<td>Game_{1,8}</td>
<td>semi-functional</td>
<td>semi-functional</td>
<td>semi-functional</td>
</tr>
<tr>
<td></td>
<td>$\alpha \leftarrow R \mathbb{Z}_p^*$</td>
<td>$\alpha \leftarrow R \mathbb{Z}_p^*$</td>
<td>$\alpha \leftarrow R \mathbb{Z}_p^*$</td>
</tr>
</tbody>
</table>

Game_{1,8}: This game is the same as Game_{1,7} except that the challenge ciphertext $ct^*$ is the semi-functional encryption of a random plaintext.

In Tables 4–6, we summarize the distributions of $ct^*$, $sk_{ID}$, $ku_{ku,T}$, $ku_{ID,T}$, and $dk_{ID,T}$ in each game. The definitions of Game_{1,0} and Game_{1,1} are identical to those of Game_{2,0} and Game_{2,1}, respectively. Thus, we can prove the indistinguishability Game_{1,0} $\approx_e$ Game_{1,1} by Lemma 3. In Game_{1,2}, we guess the value $Q^*$ with $Q_{gen}$ reduction loss. Game_{1,3} is the conceptual change that is useful to reduce the reduction loss. In Game_{1,3}, $C$ does not create $sk_{ID,T}$ and $ku_{ID,T}$ unlike in the real scheme. In turn, $C$ first creates seed secret keys $s.sk_{ID}^{(1)}$ and $s.sk_{ID}^{(2)}$, and uses the seed secret keys to create $sk_{ID,T}$ and $ku_{ID,T}$, $dk_{ID,T}$ for $T \geq T_{RL}$, respectively. For each time period $T < T_{RL}$, $C$ creates seed key update $s.ku_T$ and uses the seed key update to create $ku_{ID,T,\theta}$, $dk_{ID,T}$. In Game_{1,4}, $s.sk_{ID}^{(1)}$ revealed to $A$ become semi-functional when $ID^* \not\in ID$. We use the standard dual system argument to prove the indistinguishability Game_{1,3} $\approx_e$ Game_{1,4} (Lemma 21) by considering the fact that $ID \notin prefix^+(ID^*)$. In Game_{1,5}, $s.ku_T$ revealed to $A$ becomes semi-functional. We use the standard dual system argument to prove that all $s.ku_T$ are semi-functional by considering the fact that $T \neq T^*$. Although $s.ku_T$ are used to create $ku_{ID,T,\theta}$ and $dk_{ID,T}$, we define Game_{1,5} so that $ku_{ID,T}$ are semi-functional. Hence, we apply semi-functional randomness switching and prove the indistinguishability Game_{1,4} $\approx_e$ Game_{1,5} (Lemma 22). In Game_{1,6} and Game_{1,7}, we change $ku_{ku,T}$ and $ku_{ID,T}$, $dk_{ID,T}$ for $T \geq T_{RL}$ to be semi-functional by applying semi-functional randomness switching. Finally, in Game_{1,8}, we change the challenge ciphertext $ct^*$ to be a semi-functional encryption of a random plaintext as done in the proof against the Type-II adversary.
6.1 Proof of Lemma 1

Now, we are ready to prove Lemma 1.

**Proof of Lemma 1.** Let $\text{Adv}_i(\lambda)$ denote $\mathcal{A}$’s advantage in $\text{Game}_{\Pi,i}$. Hereafter, we prove that the difference of $\mathcal{A}$’s advantage between each game (i.e., $|\text{Adv}_{i-1}(\lambda) - \text{Adv}_i(\lambda)|$) is negligible. The indistinguishability $\text{Game}_{\Pi,0} \approx_c \text{Game}_{\Pi,1}$ is proven as Lemma 3. The key points to note is the transitions $\text{Game}_{\Pi,3} \equiv \text{Game}_{\Pi,4}$ and $\text{Game}_{\Pi,4} \approx_c \text{Game}_{\Pi,5}$ since we have to change $ku_{kgc,T}$ and $ku_{ID,T}$ such that $ID \in \text{prefix}^+(ID^*) \land T = T^*$ to be semi-functional. In other words, we rely on standard dual system proof [CGW15, CG17, CW14] to prove most of the other transitions.

**Lemma 19** ($\text{Game}_{\Pi,1} \equiv \text{Game}_{\Pi,2}$). $\text{Game}_{\Pi,1}$ and $\text{Game}_{\Pi,2}$ are identically distributed from $\mathcal{A}$’s view with non-negligible probability. Specifically, for any Type-I adversary $\mathcal{A}$ making at most $Q_{\text{gen}}$ secret key generation queries, it holds that

$$\text{Adv}_{\Pi,1}(\lambda) = Q_{\text{gen}} \cdot \text{Adv}_{\Pi,2}(\lambda).$$

**Proof of Lemma 19.** Let $E_{\Pi,1}$ and $E_{\Pi,2}$ denote the event that $\mathcal{A}$ wins in $\text{Game}_{\Pi,1}$ and $\text{Game}_{\Pi,2}$, respectively. Let $F$ denote the event that $C$’s guess is correct in $\text{Game}_{\Pi,2}$. By definition, it holds that $\Pr[F] = 1/Q_{\text{gen}}$ and $\text{Game}_{\Pi,1}$ and $\text{Game}_{\Pi,2}$ are identically distributed if $F$ happens since all the behavior of $C$ is the same. Thus, it holds that

$$\Pr[E_{\Pi,1}] = \Pr[E_{\Pi,2} \mid F]. \quad (21)$$

If $F$ does not happen, $C$ outputs a random bit and aborts the game. Thus, it holds that

$$\Pr[E_{\Pi,2} \mid \neg F] = \frac{1}{2}. \quad (22)$$

Observe that

$$\text{Adv}_{\Pi,2}(\lambda) = \left| \frac{\Pr[E_{\Pi,1}] - \frac{1}{2}}{\Pr[E_{\Pi,2} \mid F] + \Pr[E_{\Pi,2} \mid -F] - \frac{1}{2}} \right|$$

$$= \left| \frac{\Pr[E_{\Pi,1}] \cdot \Pr[F] + \Pr[E_{\Pi,1}] \cdot - \Pr[F] - \frac{1}{2}}{\Pr[E_{\Pi,1}] \cdot \Pr[F] + \Pr[E_{\Pi,1}] \cdot - \Pr[F] - \frac{1}{2}} \right|$$

From the equation (21) and (22), we have

$$\text{Adv}_{\Pi,2}(\lambda) = \left| \frac{\Pr[E_{\Pi,1}] \cdot \Pr[F] - \frac{1}{2} \cdot (1 - \Pr[-F])}{\Pr[E_{\Pi,1}] \cdot \Pr[F] - \frac{1}{2} \cdot \Pr[F]} \right|$$

$$= \left| \frac{\Pr[E_{\Pi,1}] \cdot \Pr[F] - \frac{1}{2} \cdot \Pr[F]}{\Pr[E_{\Pi,1}] \cdot \Pr[F] - \frac{1}{2} \cdot \Pr[F]} \right|$$

$$= \frac{1}{Q_{\text{gen}}} \left| \Pr[E_{\Pi,1}] - \frac{1}{2} \right|$$

$$= \frac{1}{Q_{\text{gen}}} \cdot \text{Adv}_{\Pi,1}(\lambda).$$

Thus, we complete the proof. \qed
Lemma 20 (Game$_{1,2} \equiv$ Game$_{1,3}$). Game$_{1,2}$ and Game$_{1,3}$ are identically distributed from $A$’s view. Specifically, for any PPT Type-I adversary $A$, it holds that

$$\text{Adv}_{1,2}(\lambda) = \text{Adv}_{1,3}(\lambda).$$

Proof of Lemma 20. We describe how $C$ creates $sk_{ID}$, $ku_{ID,T}$, and $dk_{ID,T}$ in Game$_{1,3}$. $C$ creates $MPK$, $sk_{T}$, and $ct^*$ in the same way as Game$_{1,2}$.

**Key Update and Decryption Key Creation for $T < T_{RL}$:** For each time period $T$ upon the setup and $A$’s revoke & key update queries, $C$ samples $t_{T} \leftarrow R Z_p^k$ and creates a seed key update $s.ku_T = (s.KU_{T,0}, s.KU_{KU,1}, s.KU_{KU,2})$ by computing (19).

For each $\theta \in KU_{NGC,T}$, $C$ retrieves a delegation key $k_{NGC,\theta}$, samples $\tilde{t}_{NGC,\theta} \leftarrow R Z_p$ and $\tilde{t}_{NGC,\theta} \leftarrow R Z_p^k$, and computes a sub-key update $k_{NGC,\theta} := (KU_{NGC,\theta,0}, KU_{NGC,\theta,1}, KU_{NGC,\theta,2})$:

$$KU_{NGC,\theta,0} = (s.KU_{T,0})^{\tilde{t}_{NGC,\theta}} \cdot \tilde{Z}_{NGC,\theta},$$

$$KU_{NGC,\theta,1} = [k - k_{NGC,\theta}]_2 \cdot (s.KU_{T,1})^{\tilde{t}_{NGC,\theta}} \cdot [([V_0 + TV_{L+1}]Z_{NGC,\theta})]_2,$$

$$KU_{NGC,\theta,2} = (s.KU_{T,2})^{\tilde{t}_{NGC,\theta}} \cdot \tilde{V}_{L+2}Z_{NGC,\theta}. \tag{23}$$

This is the normal sub-key update by setting $t_{NGC,\theta} = \tilde{t}_{NGC,\theta} \cdot t_T + \tilde{t}_{NGC,\theta}$. Due to the fresh random $\tilde{t}_{NGC,\theta} \leftarrow R Z_p^k$, $t_{NGC,\theta}$ is distributed in $Z_p^k$ uniformly at random.

For each $\theta \in KU_{NGC,T}$ such that $|ID| \geq 1$, $C$ retrieves a delegation key $k_{ID,\theta}$, samples the ephemeral delegation key $K_{ID,T} \leftarrow R Z_p^{k+1}$, $\tilde{t}_{ID,\theta} \leftarrow R Z_p$, and $t_{ID,\theta} \leftarrow R Z_p^k$, and computes a sub-key update $k_{ID,\theta} := (KU_{ID,\theta,0}, KU_{ID,\theta,1}, KU_{ID,\theta,2})$:

$$KU_{ID,\theta,0} = (s.KU_{T,0})^{\tilde{t}_{ID,\theta}} \cdot \tilde{Z}_{ID,\theta},$$

$$KU_{ID,\theta,1} = [k_{ID,\theta} + K_{ID,T}]_2 \cdot (s.KU_{T,1})^{\tilde{t}_{ID,\theta}} \cdot [([V_0 + TV_{L+1}]Z_{ID,\theta})]_2,$$

$$KU_{ID,\theta,2} = (s.KU_{T,2})^{\tilde{t}_{ID,\theta}} \cdot \tilde{V}_{L+2}Z_{ID,\theta}. \tag{24}$$

This is the normal sub-key update by setting $t_{ID,\theta} = \tilde{t}_{ID,\theta} \cdot t_T + \tilde{t}_{ID,\theta}$. Due to the fresh random $\tilde{t}_{ID,\theta} \leftarrow R Z_p^k$, $t_{ID,\theta}$ is distributed in $Z_p^k$ uniformly at random. $C$ creates the helper key update $\tilde{u}_{ID,T}$ in the same way as the real scheme.

$C$ retrieves the master secret key $k$, samples $\tilde{u}_{ID,T}' \leftarrow R Z_p$ and $u_{ID,T}', \tilde{u}_{ID,T}' \leftarrow R Z_p^k$, and computes $dk_{ID,T} = (DK_{ID,T,0}, DK_{ID,T,1}, DK_{ID,T,2}, DK_{ID,T,3})$:

$$DK_{ID,T,0} = [Z_{ID,T}]_2; \quad DK_{ID,T,1} = [k]_2 \cdot (s.KU_{T,1})^{\tilde{t}_{ID,\theta}} \cdot ([V_0 + TV_{L+1}]Z_{ID,T}]_2,$$

$$DK_{ID,T,2} = [V_{L+2}Z_{ID,T}]_2; \quad DK_{ID,T,3} = (s.KU_{T,2})^{\tilde{t}_{ID,\theta}} \cdot [V_{L+2}Z_{ID,T}]_2. \tag{25}$$

This is the normal decryption key by setting $u_{ID,T}' = \tilde{u}_{ID,T}' \cdot t_T + \tilde{u}_{ID,T}'$. Due to the fresh random $\tilde{u}_{ID,T}' \leftarrow R Z_p^k$, $u_{ID,T}'$ is distributed in $Z_p^k$ uniformly at random.

**Key Update and Decryption Key Creation for $T \geq T_{RL}$:** Upon $A$’s secret key generation query on $ID$, $C$ samples $r_{ID}^{(2)} \leftarrow R Z_p^k$ and creates a seed secret key $s.sk_{ID}^{(2)} = (s.SK_{ID,0}^{(2)}, s.SK_{ID,1}^{(2)}, s.SK_{ID,2}^{(2)})$ by computing (8). $C$ creates $k_{NGC,T}$ in the same way as the real scheme.

To create $k_{ID,T}$ such that $|ID| \geq 1$, $C$ samples the ephemeral delegation key $K_{ID,T} \leftarrow R Z_p^{k+1}$ and creates the sub-key update $k_{ID,T,\theta}$ in the same way as the real scheme. Then, $C$ retrieves the
delegation key \( k_{ID,0} \) and ephemeral delegation key \( K_{ID,T} \), samples \( \tilde{r}_{ID,T}, \tilde{r}'_{ID,T} \leftarrow_R \mathbb{Z}_p^k \), and computes a helper key update \( \tilde{K}_{ID,T} = (\tilde{K}_{ID,T,0}, \tilde{K}_{ID,T,1}, \tilde{K}_{ID,T,2}, \tilde{K}_{ID,T,3}, \tilde{K}_{ID,T,4}) \):

\[
\begin{align*}
K_{ID,T,0} &= sSK_{ID,0}^2 \cdot \tilde{z}_{ID,T}, & K_{ID,T,1} &= \Xi + \tilde{r}_{ID,T}, \\
K_{ID,T,1} &= sSK_{ID,1}^2 \cdot \left( (V_0 + i_1V_1 + \cdots + i_{|ID|}V_{|ID|}) \tilde{z}_{ID,T} \right) \tilde{r}_{ID,T}, & K_{ID,T,2} &= \Xi + \tilde{r}'_{ID,T}, \\
K_{ID,T,2} &= sSK_{ID,2}^2 \cdot \left( (V_{L+2}z_{ID,T}) \tilde{r}_{ID,T} \right) \tilde{r}'_{ID,T}, & K_{ID,T,3} &= \Xi + \tilde{r}_{ID,T}, \\
K_{ID,T,4} &= sSK_{ID,4}^2 \cdot \left( (V_{L+2}z_{ID,T}) \tilde{r}_{ID,T} \right) \tilde{r}'_{ID,T}. 
\end{align*}
\]

(26)

This is the normal helper key update as in \( Game_{1,2} \) by setting \( r_{ID,T} = r_{ID} + \tilde{r}_{ID,T} \). Due to the fresh random \( \tilde{r}_{ID,T} \leftarrow_R \mathbb{Z}_p^k \), \( \tilde{r}_{ID,T} \) is distributed in \( \mathbb{Z}_p^k \) uniformly at random.

\( C \) retrieves the master secret key \( k \), samples \( u_{ID,T}, u'_{ID,T} \leftarrow_R \mathbb{Z}_p^k \) and computes \( d_{ID,T} = (DK_{ID,T,0}, DK_{ID,T,1}, DK_{ID,T,2}, DK'_{ID,T,2}) \):

\[
\begin{align*}
DK_{ID,T,0} &= sSK_{ID,0}^2 \cdot \tilde{z}_{ID,T}, & DK'_{ID,T,0} &= \Xi + \tilde{r}'_{ID,T}, \\
DK_{ID,T,1} &= (k_2) \cdot sSK_{ID,1}^2 \cdot \left( (V_0 + i_1V_1 + \cdots + i_{|ID|}V_{|ID|}) \tilde{z}_{ID,T} \right) \tilde{r}_{ID,T}, & DK'_{ID,T,1} &= \Xi + \tilde{r}'_{ID,T}, \\
DK_{ID,T,2} &= sSK_{ID,2}^2 \cdot \left( (V_{L+2}z_{ID,T}) \tilde{r}_{ID,T} \right) \tilde{r}'_{ID,T}, & DK'_{ID,T,2} &= \Xi + \tilde{r}_{ID,T}, \\
DK_{ID,T,3} &= sSK_{ID,3}^2 \cdot \left( (V_{L+2}z_{ID,T}) \tilde{r}_{ID,T} \right) \tilde{r}'_{ID,T}. 
\end{align*}
\]

(27)

This is the normal decryption key by setting \( u_{ID,T} = r_{ID} + \tilde{u}_{ID,T} \). Due to the fresh random \( \tilde{u}_{ID,T} \leftarrow_R \mathbb{Z}_p^k \), \( u_{ID,T} \) is distributed in \( \mathbb{Z}_p^k \) uniformly at random.

As we observed so far, all the elements distribute in the same way as in \( Game_{1,2} \). Thus, we complete the proof of Lemma 20.

**Lemma 21** (Secret Key Invariance, \( Game_{1,3} \approx c Game_{1,4} \)). \( Game_{1,3} \) and \( Game_{1,4} \) are computationally indistinguishable under the MDDH assumption in \( G_2 \). Specifically, for any PPT Type-I adversary \( A \) making at most \( Q_{gen} \) secret key generation queries and \( Q_{rev} \) secret key reveal queries, there exist reduction algorithms \( B_{1,1} \) and \( B_{1,2} \) such that

\[
|Adv_{1,3}(\lambda) - Adv_{1,4}(\lambda)| \leq Q_{rev} \cdot \sum_{j\in[2]} \text{Adv}_{B_{1,j}}^{MDDH-\mathbb{G}_2}(\lambda) + \frac{4Q_{rev}}{p-1}
\]

and \( \max_{j\in[2]} T(B_{1,j}) \approx T(A) + k^2Q_{gen}|T| \cdot \text{poly}(\lambda, L) \), where \( \text{poly}(\lambda, L) \) is independent of \( T(A) \).

We omit the proof of Lemma 21 since it is essentially the same as the proof of Lemma 5. The only essential difference is that \( sSK_{ID} \) such that \( IDQ^* \in \text{prefix}^+(ID) \) are always normal by computing (8).

Since \( A \) makes secret key reveal queries on \( ID \) such that \( IDQ^* \in \text{prefix}^+(ID) \) only after \( A \)'s secret key generation query on \( IDQ^* \), the reduction algorithm can detect whether \( IDQ^* \in \text{prefix}^+(ID) \) holds.

**Lemma 22** (Key Update and Decryption Key Invariance for \( T < T_{RL}, Game_{1,4} \approx c Game_{1,5} \)). \( Game_{1,4} \) and \( Game_{1,5} \) are computationally indistinguishable under the MDDH assumption in \( G_2 \). Specifically, for any PPT Type-I adversary \( A \) making at most \( Q_{gen} \) secret key generation queries, there exist reduction algorithms \( B_{1,3} \) and \( B_{1,4} \) such that

\[
|Adv_{1,4}(\lambda) - Adv_{1,5}(\lambda)| \leq T_{RL} \cdot \sum_{j\in[2]} \text{Adv}_{B_{1,j}}^{MDDH-\mathbb{G}_2}(\lambda) + \frac{4T_{RL}}{p-1}
\]

and \( \max_{j\in[2]} T(B_{1,j+2}) \approx T(A) + k^2Q_{gen}|T| \cdot \text{poly}(\lambda, L) \), where \( \text{poly}(\lambda, L) \) is independent of \( T(A) \).
Proof of Lemma 20. To prove Lemma 20, we further introduce the following auxiliary distributions.

Pseudo-normal Seed Key Updates: A pseudo-normal seed key update is defined as s.ku_T := (s.KU_T,0, s.KU_T,1, s.KU_T,2):

\[ s.KU_T,0 := [Zt_T]_2, \quad s.KU_T,1 := [(V_0 + TV L+1)Zt_T]_2 \cdot \left[ (a^1)^0 + v_{L+1}^T \right], \]

\[ s.KU_T,2 := [V_{L+2}Zt_T]_2 \cdot \left( [a^1]^{-1} \right), \]

where \( t_{ID,T} \leftarrow R \mathbb{Z}_p^k \), \( i \leftarrow R \mathbb{Z}_p^* \), and \( (v_0, v_{L+1}) \leftarrow R \mathbb{Z}_p^2 \) is the randomness for creating the challenge ciphertext. Here, the boxed parts denote the changes from the normal seed key update.

Pseudo-SF Seed Key Updates: A pseudo-SF seed key update is defined as s.ku_T := (s.KU_T,0, s.KU_T,1, s.KU_T,2):

\[ s.KU_T,0 := [Zt_T]_2, \quad s.KU_T,1 := [(\alpha a^1)^0]_2 \cdot [(V_0 + TV L+1)Zt_T]_2 \cdot \left[ (a^1)^0 + v_{L+1}^T \right], \]

\[ s.KU_T,2 := [V_{L+2}Zt_T]_2 \cdot \left( [a^1]^{-1} \right), \]

where \( t_{ID,T} \leftarrow R \mathbb{Z}_p^k \), \( i \leftarrow R \mathbb{Z}_p^* \), \( (v_0, v_{L+1}) \leftarrow R \mathbb{Z}_p^2 \) is the randomness for creating the challenge ciphertext, and \( \alpha \leftarrow R \mathbb{Z}_p^* \) is the semi-functional randomness shared by all s.sk^{(1)} and s.ku_T. Here, the boxed part denotes the change from the pseudo-normal seed key update.

We further introduce the following sequence of games for \( T \in [0, TRL - 1] \):

**Game_{1,4,T,1}:** This game is the same as Game_{1,3} except that
- If \( T < T \), \( C \) creates semi-functional s.ku_T upon \( A \)'s secret key generation queries,
- If \( T = T \), \( C \) creates pseudo-normal s.ku_T upon \( A \)'s secret key generation queries,
- If \( T > T \), \( C \) always creates normal s.ku_T upon \( A \)'s secret key generation queries, secret key generation queries.

**Game_{1,4,T,2}:** This game is the same as Game_{1,4,T,1} except that
- If \( T = T \), \( C \) creates pseudo-SF s.ku_T upon \( A \)'s secret key generation queries,

**Game_{1,4,T,3}:** This game is the same as Game_{1,4,T,2} except that
- If \( T = T \), \( C \) creates semi-functional s.ku_T upon \( A \)'s secret key generation queries,

By definition, Game_{1,4,0,3} = Game_{1,4}. Hereafter, we prove

\[ \text{Game}_{1,4,T-1,3} \approx \text{Game}_{1,4,T-1,1} \equiv \text{Game}_{1,4,T-1,2} \equiv \text{Game}_{1,4,T,3}, \]

where the fact implies that Game_{1,4} \approx \text{Game}_{1,4,TRL-1,3}. We note that Game_{1,4,TRL-1,3} \equiv \text{Game}_{1,5} will be proved later.

Lemma 23 (Seed Key Updates Transition from Normal to Pseudo-normal, Game_{1,4,T-1,3} \approx \text{Game}_{1,4,T,1}). Game_{1,4,T-1,3} and Game_{1,4,T,1} are computationally indistinguishable under the MDDH assumption in \( \mathbb{G}_2 \). Specifically, for any PPT Type-I adversary \( A \) making at most \( Q_{gen} \) secret key generation queries, there exists a reduction algorithm \( B_{1,3} \) such that

\[ |\text{Adv}_{1,4,T-1,3}(\lambda) - \text{Adv}_{1,4,T,1}(\lambda)| \leq \text{Adv}_{B_{1,3}}^{\text{MDDH-G}_2}(\lambda) + \frac{2}{p - 1}, \]

and \( T(B_{1,3}) \approx T(A) + k^2 Q_{gen} |T| \cdot \text{poly}(\lambda, L) \), where \( \text{poly}(\lambda, L) \) is independent of \( T(A) \).
Proof of Lemma 23. The reduction algorithm $B_{1,3}$ is given a MDDH instance in $G_2$: $(G(1^\lambda), [b]_2, [b]_2 = [Bt + te]_2)$, where $B \xleftarrow{R} D_k$, $t \xleftarrow{R} Z_p^k$, $\hat{t} = 0$ or $\hat{t} \xleftarrow{R} Z_p$, and $e = (0, \ldots, 0, 1)^T \in Z_p^{k+1}$. Hereafter, we assume that $\hat{t} \xleftarrow{R} Z_p^*$ in the latter case with the statistical difference $1/p$.

$B_{1,3}$ creates MPK and $ct^*$ in the same way as the proof of Lemma 6. $B_{1,3}$ creates semi-functional $s,sk^{(1)}_\text{ID}$ by computing (9) and creates $sk_\text{ID}$ by computing (10). $B_{1,3}$ creates $ku_{\text{ID},T}$ and $dk_{\text{ID},T}$ for $T \geq T_{RL}$ in the same way as the proof of Lemma 20.

After creating $s,ku_T$, $B_{1,3}$ creates $ku_{\text{ID},T}$ and $dk_{\text{ID},T}$ for $T \geq T_{RL}$ in the same way as the proof of Lemma 20. We describe how $B_{1,3}$ creates $s,ku_T = (s,\text{KU}_{T,0},s,\text{KU}_{T,1},s,\text{KU}_{T,2})$.

- If $T < T$, $B_{1,3}$ creates semi-functional $s,ku_T$ by computing (20).
- If $T = T$, $B_{1,3}$ retrieves $(v_0, v_{L+1})$ and computes $s,ku_T = (s,\text{KU}_{T,0},s,\text{KU}_{T,1},s,\text{KU}_{T,2})$:

\[
s,\text{KU}_{T,0} = \overline{B}_2, \quad s,\text{KU}_{T,1} = [(\overline{V}_0 + TV_{L+1})\overline{B}]_2 : [a^t b]^{v_0 + v_{L+1} \epsilon}, \quad s,\text{KU}_{T,2} = [V_{L+2}\overline{B}]_2 : [a^t b]. \tag{28}
\]

By following the same argument in the proof of Lemma 6, $s,ku_T$ is a normal seed key update as in Game$_{1,4,T-1,3}$ if $\hat{t} = 0$, and pseudo-normal seed key update as in Game$_{1,4,T,1}$ if $\hat{t} \xleftarrow{R} Z_p^*$, by setting $t_T = \overline{Z}^{-1} t$.

- If $T > T$, $B_{1,3}$ creates normal $s,ku_T$ by computing (19).

Thus, we complete the proof of Lemma 23. 

\[\square\]

Lemma 24 (Seed Key Update Transition from Pseudo-normal to Pseudo-SF, Game$_{1,4,T,1} \equiv$ Game$_{1,4,T,2}$). Game$_{1,4,T,1}$ and Game$_{1,4,T,2}$ are identically distributed from $A$’s view. Specifically, for any Type-I adversary $A$, it holds that

\[\text{Adv}_{1,4,T,1}(\lambda) = \text{Adv}_{1,4,T,2}(\lambda).\]

We can prove that $s,ku_T$ follows the same distribution in Game$_{1,4,T,1}$ and Game$_{1,4,T,2}$ by following the same argument as in the proof of Lemma 7 based on the fact that $T \neq T^*$ for all $T < T_{RL}$.

Lemma 25 (Seed Key Update Transition from Pseudo-SF to Semi-functional, Game$_{1,4,T,2} \approx_c$ Game$_{1,4,T,3}$). Game$_{1,4,T,2}$ and Game$_{1,4,T,3}$ are computationally indistinguishable under the MDDH assumption in $G_2$. Specifically, for any PPT Type-I adversary $A$ making at most $Q_{\text{gen}}$ secret key generation queries, there exists a reduction algorithm $B_{1,4}$ such that

\[|\text{Adv}_{1,4,T,2}(\lambda) - \text{Adv}_{1,4,T,3}(\lambda)| \leq \text{Adv}_{B_{1,4}}^{\text{MDDH}G_2}(\lambda) + \frac{2}{p-1}\]

and $T(B_{1,4}) \approx T(A) + k^2Q_{\text{gen}}|T| \cdot \text{poly}(\lambda, L)$, where $\text{poly}(\lambda, L)$ is independent of $T(A)$.

We omit the detailed proof of Lemma 25 since it is almost the same as the proof of Lemma 23. The only difference is that $B_{1,4}$ creates $T$-th $s,ku_T$ by computing (28) except that

\[s,\text{KU}_{T,1} = ([a^t b] \cdot [(\overline{V}_0 + TV_{L+1})\overline{B}]_2 : [a^t b]^{v_0 + v_{L+1} \epsilon}, \quad \overline{V}_{L+2}\overline{B}.\]

where the boxed parts denote the changes from (28). If $\hat{t} \xleftarrow{R} Z_p^*$, $s,ku_T$ is a pseudo-SF seed key update as in Game$_{1,4,T,2}$. If $\hat{t} = 0$, $s,ku_T$ is a semi-functional seed key update as in Game$_{1,4,T,3}$.
Lemma 26 (Semi-functional Randomness Switching for Helper Key Updates for $T < T_{RL}$, $\text{Game}_{1,4,T_{RL}-1,3} = \text{Game}_{1,5}$). $\text{Game}_{1,4,T_{RL}-1,3}$ and $\text{Game}_{1,5}$ are identically distributed from $A$'s view. Specifically, for any Type-I adversary $A$, it holds that

$$\text{Adv}_{1,4,T_{RL}-1,3}(\lambda) = \text{Adv}_{1,5}(\lambda).$$

The proof is the first core part of the proof against the Type-I adversary. In $\text{Game}_{1,4,T_{RL}-1,3}$, all the seed key updates $s.k_u T$ for $T < T_{RL}$ become semi-functional. Although we use $s.k_u T$ to create $k_{ID,T,\theta}$ and $d_{k_{ID,T}}$, we defined $\text{Game}_{1,5}$ so that all $k_{ID,T}$ for $T < T_{RL}$ to be semi-functional. For this purpose, we apply the semi-functional randomness switching to show that $\text{Game}_{1,4,T_{RL}-1,3} \equiv \text{Game}_{1,5}$.

Proof of Lemma 26. Here, we prove a stronger claim that $\text{Game}_{1,4,T_{RL}-1,3}$ and $\text{Game}_{1,5}$ are identically distributed from $A$'s view for any fixed

- $(A, a) \leftarrow R D_k$,
- $((V\ell)_{\ell \in [0,L-2]}, Z) \leftarrow R (Z_p^{(k+1) \times k})^{L+3} \times Z_p^{L \times k}$,
- master secret key $k \leftarrow R Z_p^{k+1}$,
- $t_{ID,T,\theta} \leftarrow R Z_p^k$ for creating $k_{ID,T,\theta}$,
- $\overline{t}_{ID,T} \leftarrow R Z_p^k$ for creating $\overline{k}_{ID,T}$,
- $\alpha \leftarrow R Z_p^2$.

Specifically, the randomnesses of $\overline{t}_{ID,T,\theta} \leftarrow R Z_p$ in (24) and $\overline{\text{del}}k_{ID,T} \leftarrow R Z_p^{k+1}$ enable us to prove the claim. Note that even when $t_{ID,T,\theta} \leftarrow R Z_p^k$ are fixed, $(K_{U_{ID,T,\theta},0}, K_{U_{ID,T,\theta},2})$ do not reveal the quantities of $\overline{t}_{ID,T,\theta} \leftarrow R Z_p$ since they are masked by $\overline{t}_{ID,T,\theta} \leftarrow R Z_p^k$. Since $t_{ID,T,\theta} \leftarrow R Z_p^k$ and $\overline{t}_{ID,T} \leftarrow R Z_p^k$ are fixed, $(K_{U_{ID,T,\theta},0}, K_{U_{ID,T,\theta},2})$ and $(\overline{K}_{U_{ID,T,0}}, \overline{K}_{U_{ID,T,2}})$ follow the same distribution in both $\text{Game}_{1,4,T_{RL}-1,3}$ and $\text{Game}_{1,5}$. In $\text{Game}_{1,4,T_{RL}-1,3}$, $K_{U_{ID,T,\theta},1}$ and $\overline{K}_{U_{ID,T,1}}$ for $T < T_{RL}$ such that $|ID| \geq 1$ are distributed as follows:

$$K_{U_{ID,T,\theta},1} = [k_{ID,θ} + \overline{k}_{ID,T} - \overline{t}_{ID,T,θ} - 1] \cdot [(V_0 + TV_{L+1})Z_{t_{ID,T,θ}}]_2,$$

$$\overline{K}_{U_{ID,T,1}} = [k + \overline{k}_{ID,T}]_2 \cdot [(V_0 + id_{1}V_1 + \cdots + id_{|ID|}V_{|ID|})Z_{\overline{t}_{ID,T}}]_2 \cdot [(V_0 + TV_{L+1})Z_{\overline{t}_{ID,T}}]_2,$$

where $\overline{t}_{ID,T,θ} \leftarrow R Z_p^k$ and $\overline{\text{del}}k_{ID,T} = \overline{k}_{ID,T} \leftarrow R Z_p^{k+1}$. In contrast, the above distribution can be written as follows:

$$K_{U_{ID,T,\theta},1} = [k_{ID,θ} + (\overline{k}_{ID,T} - \overline{t}_{ID,T,θ}) - (\overline{t}_{ID,T,θ} - 1)αa]_2,$$

$$\overline{K}_{U_{ID,T,1}} = [(k + αa^2)]_2 \cdot [(V_0 + TV_{L+1})Z_{\overline{t}_{ID,T}}]_2 \cdot [(V_0 + TV_{L+1})Z_{\overline{t}_{ID,T}}]_2,$$

where $\overline{t}_{ID,T,θ} - 1$ is distributed in $Z_p$ uniformly at random and $\overline{k}_{ID,T} - αa$ is distributed in $Z_p^{k+1}$ uniformly at random. Therefore, the above distribution is the same as the distribution in $\text{Game}_{1,5}$ by setting $\overline{t}_{ID,T,θ} - 1$ as the randomnesses in (10) and $\overline{\text{del}}k_{ID,T} = \overline{k}_{ID,T} + αa$. We note that the claim holds for all ID such that $|ID| \geq 1$, all $T < T_{RL}$, and all nodes $θ \in BT_{ID}$, simultaneously. Thus, we complete the proof of Lemma 26.

By combining Lemmata 23–26, we have

$$|\text{Adv}_{1,4}(\lambda) - \text{Adv}_{1,5}(\lambda)|$$
Thus, we complete the proof of Lemma 22.

**Lemma 27** (Semi-functional Randomness Switching for KGC’s Key Updates for $T \geq T_{RL}$, $\text{Game}_{1,5} \equiv \text{Game}_{1,6}$). $\text{Game}_{1,5}$ and $\text{Game}_{1,6}$ are identically distributed from $A$’s view. Specifically, for any Type-I adversary $A$, it holds that

$$\text{Adv}_{1,5}(\lambda) = \text{Adv}_{1,6}(\lambda).$$

The proof is the second core part of the proof against the Type-I adversary since we have to change $ku_{kgc,T}$ to be semi-functional. We want to discuss the difference from the proof of Lemma 9. In the proof of Lemma 9, all $sk_{ID}$ such that $|ID| = 1$ which $A$ receives via secret key *reveal* queries are semi-functional. In contrast, in the proof of Lemma 27, $A$ may receive $sk_{ID[\ell*]} = sk_{ID[i]}$ when $\ell^* = 1$. However, once $A$ receives $sk_{ID[1]}^* ID_1^*$ must be revoked at $T_{RL}$. In other words, $sk_{ID[1]}$ and $ku_{kgc,T}$ for $T \geq T_{RL}$ do not share the same nodes $\theta \in \mathcal{B}_{kgc}$. The fact is sufficient for proving Lemma 27 by combining with the modifications so far.

**Proof of Lemma 27.** Here, we prove a stronger claim that $\text{Game}_{1,5}$ and $\text{Game}_{1,6}$ are identically distributed from $A$’s view for any fixed

- $(A,a) \leftarrow R \mathcal{D}_k$,
- $((V_i)_{i \in [0:L+2]}, Z) \leftarrow R (Z_p^{k+1}) \times Z_p^k$,
- master secret key $k \leftarrow R \mathbb{Z}_p^{k+1}$,
- $r_{ID,\theta} \leftarrow R \mathbb{Z}_p^k$ for creating $sk_{ID,\theta}$ such that $|ID| = 1$,
- $t_{kgc,T,\theta} \leftarrow R \mathbb{Z}_p^k$ for creating $ku_{kgc,T}$,
- $\alpha \leftarrow R \mathbb{Z}_p^k$ that is the semi-functional randomness of $ku_{kgc,T}$ in $\text{Game}_{1,6}$.

Specifically, the randomnesses of $r_{ID,\theta} \leftarrow R \mathbb{Z}_p^k$ in (10), $t_{kgc,T,\theta} \leftarrow R \mathbb{Z}_p^k$ in (23), and $del_{kgc,\theta} \leftarrow R \mathbb{Z}_p^{k+1}$ enable us to prove the claim. Note that $sk_{ID}$ such that $|ID| \geq 2$, $ku_{ID,T}$ such that $|ID| \geq 1$, and $dk_{ID,T}$ are created in the same way in both $\text{Game}_{1,5}$ and $\text{Game}_{1,6}$. Since $r_{ID,\theta}$ and $t_{kgc,T,\theta}$ are fixed, $sk_{ID,\theta}$ such that $|ID| = 1$ and $ku_{kgc,T,\theta}$ are distributed in the same way in both $\text{Game}_{1,5}$ and $\text{Game}_{1,6}$ except $SK_{ID,\theta,1}$ and $KU_{kgc,T,\theta,1}$. Note that even when $r_{ID,\theta} \leftarrow R \mathbb{Z}_p^k$ and $t_{kgc,T,\theta} \leftarrow R \mathbb{Z}_p^k$ are fixed, $(SK_{ID,\theta,0}, SK_{ID,\theta,2}, (SK_{ID,\theta,\ell})_{\ell \in [|ID|+1,|L|])}$ and $(KU_{kgc,T,\theta,0}, KU_{kgc,T,\theta,2})$ do not reveal the quantities of $r_{ID,\theta} \leftarrow R \mathbb{Z}_p^k$ and $t_{kgc,T,\theta} \leftarrow R \mathbb{Z}_p^k$ since they are masked by $r_{ID,\theta} \leftarrow R \mathbb{Z}_p^k$ and $t_{kgc,T,\theta} \leftarrow R \mathbb{Z}_p^k$, respectively. In $\text{Game}_{1,5}$, for all nodes $\theta \in \mathcal{B}_{kgc}$ that correspond to $ku_{kgc,T,\theta}$ for $T \geq T_{RL}$, $SK_{ID,\theta,1}$ and $KU_{kgc,T,\theta,1}$ are distributed as follows:

$$SK_{ID,\theta,1} = [k_{kgc,\theta} + r_{ID,\theta,\theta} \alpha^{-1}] \cdot [(V_0 + IDV_1)Z_{r_{ID,\theta}}],$$

$$KU_{kgc,T,\theta,1} = [k - k_{kgc,\theta} + t_{kgc,T,\theta} \alpha^{-1}] \cdot [(V_0 + TV_{L+1})Z_{r_{kgc,T,\theta}}]$$

for $T < T_{RL}$,

$$KU_{kgc,T,\theta,1} = [k - k_{kgc,\theta}] \cdot [(V_0 + TV_{L+1})Z_{r_{kgc,T,\theta}}]$$

for $T \geq T_{RL}$,

where $r_{ID,\theta} \leftarrow R \mathbb{Z}_p^k$, $t_{kgc,T,\theta} \leftarrow R \mathbb{Z}_p^k$, and $del_{kgc,\theta} = k_{kgc,\theta} \leftarrow R \mathbb{Z}_p^{k+1}$. As we observed above, the quantities of $r_{ID,\theta} \leftarrow R \mathbb{Z}_p^k$ and $t_{kgc,T,\theta} \leftarrow R \mathbb{Z}_p^k$ are revealed to $A$ only via $SK_{ID,\theta,1}$ and $KU_{kgc,T,\theta,1}$. 

52
Although sk_{ID_1}^{T, \theta}, which is created by the normal s.sk_{ID_1}^{(1)}, may be revealed to A, they do not share the same nodes with ku_{kgc,T, \theta} for T \geq T_{RL} since ID_{1}^{*} is revoked by T_{RL} and the property of the CS method ensures the fact. Thus, all sk_{ID, \theta} that share the same nodes with ku_{kgc,T, \theta} are created by the semi-functional s.sk_{ID}^{(1)} as we specified above.

In contrast, the above distribution can be written as follows:

\[ SK_{ID, \theta, 1} = [(k_{kgc, \theta} + \alpha a_{V}) + (\tilde{r}_{ID, \theta} - 1)\alpha a_{V}]_{2} \cdot [(V_{0} + ID_{1}V_{1})Z_{r_{ID, \theta}}]_{2}, \]
\[ KU_{kgc,T, \theta, 1} = [k - (k_{kgc, T} + \alpha a_{V}) + (\tilde{t}_{kgc, T, \theta} + 1)\alpha a_{V}]_{2} \cdot [(V_{0} + TV_{L+1})Z_{t_{kgc, T, \theta}}]_{2} \quad \text{for } T < T_{RL}, \]
\[ KU_{kgc,T, \theta, 1} = [(k + \alpha a_{V}) - (k_{kgc, T} + \alpha a_{V})]_{2} \cdot [(V_{0} + TV_{L+1})Z_{t_{kgc, T, \theta}}]_{2} \quad \text{for } T \geq T_{RL}, \]

where \( \tilde{r}_{ID, \theta} - 1 \) and \( \tilde{t}_{kgc, T, \theta} + 1 \) are distributed in \( Z_{p} \) uniformly at random and \( k_{kgc, \theta} + \alpha a_{V} \) is distributed in \( Z_{p}^{k+1} \) uniformly at random. Therefore, the above distribution is the same as the distribution in Game_{1,6} by setting \( \tilde{r}_{ID, \theta} - 1 \) and \( \tilde{t}_{kgc, T, \theta} + 1 \) as the randomnesses in (10) and (23), respectively, and \( del_{kgc, \theta} = k_{kgc, \theta} + \alpha a_{V} \). We note that the claim holds for all \( T \geq T_{RL} \) and all nodes \( \theta \) that correspond to \( ku_{kgc,T, \theta} \) for \( T \geq T_{RL} \), simultaneously. Thus, we complete the proof of Lemma 27.

\[ SK_{ID, \theta, 1} = [(k_{kgc, \theta} + \alpha a_{V}) + (\tilde{r}_{ID, \theta} - 1)\alpha a_{V}]_{2} \cdot [(V_{0} + ID_{1}V_{1})Z_{r_{ID, \theta}}]_{2}, \]
\[ KU_{kgc,T, \theta, 1} = [k - (k_{kgc, T} + \alpha a_{V}) + (\tilde{t}_{kgc, T, \theta} + 1)\alpha a_{V}]_{2} \cdot [(V_{0} + TV_{L+1})Z_{t_{kgc, T, \theta}}]_{2} \quad \text{for } T < T_{RL}, \]
\[ KU_{kgc,T, \theta, 1} = [(k + \alpha a_{V}) - (k_{kgc, T} + \alpha a_{V})]_{2} \cdot [(V_{0} + TV_{L+1})Z_{t_{kgc, T, \theta}}]_{2} \quad \text{for } T \geq T_{RL}, \]

Lemma 28 (Key Update and Decryption Key Invariance for \( |ID| \geq 1 \) and \( T \geq T_{RL} \), Game_{1,6} \approx_{c} Game_{1,7}). Game_{1,6} and Game_{1,7} are computationally indistinguishable under the MDDH assumption in \( \mathbb{G}_2 \). Specifically, for any PPT Type-I adversary A making at most \( Q_{gen} \) secret key generation queries, there exists reduction algorithms \( B_{1,5} \) and \( B_{1,6} \) such that

\[ |Adv_{1,6}(\lambda) - Adv_{1,7}(\lambda)| \leq Q_{gen} \cdot \sum_{j \in [2]} Adv_{B_{1,5+j}}^{MDDH-\mathbb{G}_2}(\lambda) + \frac{4Q_{gen}}{p - 1}, \]

and \( \max_{j \in [2]} T(B_{1,5+j}) \approx T(A) + k^{2}Q_{gen}|T| \cdot \text{poly}(\lambda, L) \), where \( \text{poly}(\lambda, L) \) is independent of \( T(A) \).

The structure of the proof is the same as the proof of Lemma 5 although the transition from pseudo-normal to pseudo-SF is more technical.

Proof of Lemma 28. Let ID_{q} denote an identity on which A makes \( q \)-th secret key generation query. We further introduce the following sequence of games for \( q \in [0, Q_{gen}] \):

Game_{1,6,q,1}: This game is the same as Game_{1,6} except that
- If \( m < q \), C creates semi-functional \( ku_{ID_{m}} \) and \( d_{kID_{m}} \).
- If \( m = q \), C creates pseudo-normal s.sk_{ID_{q}}^{(2)} upon A’s \( q \)-th secret key generation query,
- If \( m > q \), C creates normal s.sk_{ID_{m}}^{(2)} upon A’s last \( Q_{gen} - q \) secret key generation queries.

Game_{1,6,q,2}: This game is the same as Game_{1,6,q,1} except that
- If \( m < q \), C creates pseudo-SF s.sk_{ID_{q}}^{(2)} upon A’s \( q \)-th secret key generation query,

Game_{1,6,q,3}: This game is the same as Game_{1,6,q,2} except that
- If \( m = q \), C creates semi-functional s.sk_{ID_{q}}^{(2)} upon A’s \( q \)-th secret key generation query.

By definition, Game_{1,6,0,3} = Game_{1,6} and Game_{1,6,Q_{gen},3} = Game_{1,7}. Hereafter, we prove

\[ Game_{1,6,q-1,q} \approx_{c} Game_{1,6,q} \equiv Game_{1,6,q} \approx_{c} Game_{1,6,q+1}, \]

where the fact implies that Game_{1,6} \approx_{c} Game_{1,7}. 53
Lemma 29 (Sub-secret Key Transition from Normal to Pseudo-normal, Game$_{1,6,q-1,3}$ $\approx$ Game$_{1,6,q,1}$). Game$_{1,6,q-1,3}$ and Game$_{1,6,q,1}$ are computationally indistinguishable under the MDDH assumption in $\mathbb{G}_2$. Specifically, for any PPT Type-I adversary $A$ making at most $Q_{\text{gen}}$ secret key generation queries, there exists a reduction algorithm $B_{1.5}$ such that

$$|\text{Adv}_{1,6,q-1,3}(\lambda) - \text{Adv}_{1,6,q,1}(\lambda)| \leq \text{Adv}_{B_{1.5}}^{\text{MDDH-G}_2}(\lambda) + \frac{2}{p-1}$$

and $T(B_{1.5}) \approx T(A) + k^2Q_{\text{gen}}|T| \cdot \text{poly}(\lambda, L)$, where $\text{poly}(\lambda, L)$ is independent of $T(A)$.

We omit the proof since it is almost the same as the proof of Lemma 6.

Lemma 30 (Sub-secret Key Transition from Pseudo-normal to Pseudo-SF, Game$_{1,6,q,1}$ $\equiv$ Game$_{1,6,q,2}$). Game$_{1,6,q,1}$ and Game$_{1,6,q,2}$ are identically distributed from $A$'s view. Specifically, for any Type-I adversary $A$, it holds that

$$\text{Adv}_{1,6,q,1}(\lambda) = \text{Adv}_{1,6,q,2}(\lambda).$$

The proof of Lemma 30 is the final core part of the proof against the Type-I adversary since we have to change all $ku_{ID,T}$ and $dk_{ID,T}$ such that $ID_q \in \text{prefix}^+(ID_{[\ell^s,-1]})$ to be semi-functional. We note that since $ID_{[\ell^s]}$ is revoked at the time period $T_{RL}$ we do not have to change $ku_{ID,T}$ and $dk_{ID,T}$ such that $ID_{[\ell^s]} \in \text{prefix}^+(ID)$ to be semi-functional. When it holds that $ID_q \notin \text{prefix}^+(ID_{[\ell^s,-1]})$, we prove Lemma 30 in the same way as the proof of Lemma 12 by showing that pseudo-normal and pseudo-SF $s.sk_{ID_q}^{(2)}$ are identically distributed. When $ID_q \in \text{prefix}^+(ID_{[\ell^s,-1]})$, we also follow the same argument as in the proof of Lemma 12 by applying the semi-functional randomness switching. In addition, as the proof of Lemma 27, we use the fact that although all $sk_{ID,\theta}$ such that $pa(ID) = ID_q$ may be created by the normal $s.sk_{ID}^{(1)}$, all $sk_{ID,\theta}$ that share the same nodes with $ku_{ID_q,T,\theta}$ for $T \geq T_{RL}$ are created by the semi-functional $s.sk_{ID}^{(1)}$.

Proof of Lemma 30. If $ID_q = (id_{q,1}, \ldots, id_{q,ID_q}) \notin \text{prefix}^+(ID_{[\ell^s,-1]})$, we can show that pseudo-normal and pseudo-SF $s.sk_{ID_q}^{(2)}$ are identically distributed by following the same argument as in the proof of Lemma 7.

If $ID_q = (id_{q,1}, \ldots, id_{q,ID_q}) \in \text{prefix}^+(ID_{[\ell^s,-1]})$, we prove a stronger claim that Game$_{1,6,q,1}$ and Game$_{1,6,q,2}$ are identically distributed from $A$’s view for any fixed

- $(A, a) \leftarrow_R D_k$,
- $((V_\ell)_{\ell \in [0,L+2]}, Z) \leftarrow_R (Z_p^{(k+1)\times k})^{L+3} \times Z_p^{k \times k}$,
- master secret key $k \leftarrow_R Z_p^{k + 1}$,
- $r_{ID,\theta} \leftarrow_R Z_p^k$ for creating $sk_{ID,\theta}$ such that $pa(ID) = ID_q$,
- $\overline{t}_{ID_q,T} \leftarrow_R Z_p^k$ for creating $ku_{ID_q,T}$ for $T < T_{RL}$,
- $t_{ID_q,T,\theta}, \overline{t}_{ID_q,T} \leftarrow_R Z_p^k$ for creating $ku_{ID_q,T}$,
- $u_{ID_q,T}, u_{ID_q,T}^\ast \leftarrow_R Z_p^k$ for creating $dk_{ID_q,T}$,
- $\alpha \leftarrow_R Z_p^\ast$.

Note that $sk_{ID}$ such that $pa(ID) \neq ID_q$, $ku_{ID,T}$ and $dk_{ID,T}$ such that $ID \neq ID_q$ are created in the same way in both Game$_{1,6,q,1}$ and Game$_{1,6,q,2}$.

At first, we show that the randomness of $(v_0, v_1, \ldots, v_{|ID^s|}, v_{L+1}) \leftarrow_R Z_p^{|ID^s|+2}$ enables us to prove that all $dk_{ID_q,T}$ created by using $s.sk_{ID_q}^{(2)}$ follow the same distribution in Game$_{1,6,q,1}$ and Game$_{1,6,q,2}$.
For this purpose, it is sufficient to show that \( (s \cdot SK_{ID_q}^{(2)}(\theta_0) \cdot s \cdot SK_{ID_q}^{(2)}(\theta_1) \cdot s \cdot SK_{ID_q}^{(2)}(\theta_2) ) \) follows the same distribution in \( Game_{1,6,q,1} \) and \( Game_{1,6,q,2} \) since \( (s \cdot SK_{ID_q}^{(2)}(\theta) )_{\ell \in [1, |ID_q| + 1, L]} \) is not used for creating \( dk_{ID_q,T} \). By following the same argument in the proof of Lemma 7, what we have to show is

\[
\begin{align*}
\left\{ \begin{array}{l}
v_0 + v_1 id_1^* + \cdots + v_{|ID_q|} id_{|ID_q|}^*; v_0 + v_{L+1} T^*; \\
v_0 + v_1 id_{q,1} + \cdots + v_{|ID_q|} id_{|ID_q|}^*;\end{array} \right.
\equiv \left\{ \begin{array}{l}
v_0 + v_1 id_1^* + \cdots + v_{|ID_q|} id_{|ID_q|}^*; v_0 + v_{L+1} T^*; \\
\alpha/\hat{r} + v_0 + v_1 id_{q,1} + \cdots + v_{|ID_q|} id_{|ID_q|}^*;\end{array} \right.
\end{align*}
\]

(29)

where \( (v_0, v_1, \ldots, v_{|ID_q|}, v_{L+2}) \leftarrow R Z_p^{ID_q^*+2} \). Here, the first and second elements are \( tag \) and \( tag' \) and the last element is the exponent of \( \hat{r} a_q^{-1} \) of \( s \cdot SK_{ID_q}^{(2)}(\theta_1) \) in \( Game_{1,6,q,1} \) and \( Game_{1,6,q,2} \), respectively. Since the only second element depends on \( v_{L+1} \leftarrow R Z_p \), the second element is distributed in \( Z_p \) uniformly at random. As we observed above, it holds that \( |ID_q| < l^* \leq |ID_q^*| \). Thus, since the only first element depends on \( \langle v_{|ID_q|+1}, \ldots, v_{|ID_q|} \rangle \leftarrow R Z_p^{ID_q^*-[|ID_q|]+1} \), the first element is distributed in \( Z_p \) uniformly at random. As a result, the last element is also distributed in \( Z_p \) uniformly at random. Summarizing the discussion so far, both hand sides of \( (29) \) are distributed in \( Z_p^2 \) uniformly at random. Thus, we complete the proof of the claim that all \( dk_{ID_q,T} \) created by using \( s \cdot sk_{ID_q}^{(2)} \) follow the same distribution in \( Game_{1,6,q,1} \) and \( Game_{1,6,q,2} \).

Finally, we show that for any fixed

- \( (v_0, v_1, \ldots, v_{L+1}) \leftarrow R Z_p^{L+2} \),

the randomness of \( \hat{r}_{ID_q} \leftarrow R Z_p \) such that \( pa(ID) = ID_q \) in \( (10) \), \( \hat{t}_{ID_q,T,\theta} \leftarrow R Z_p \) in \( (23) \), and \( \hat{d}elk_{ID_q,\theta}, \hat{d}elk_{ID_q,T} \leftarrow R Z_p^{k+1} \) for \( T \geq T_{RL} \) enable us to prove that all \( ku_{ID_q,T} \) created by using \( s \cdot sk_{ID_q}^{(2)} \) follow the same distribution in \( Game_{1,6,q,1} \) and \( Game_{1,6,q,2} \). Since \( r_{ID_q,\theta} \) and \( t_{ID_q,T,\theta} \) are fixed, \( sk_{ID_q} \) such that \( pa(ID) = ID_q \) and \( ku_{ID_q,T} \) for \( T \geq T_{RL} \) are distributed in the same way in both \( Game_{1,6,q,1} \) and \( Game_{1,6,q,2} \) except \( SK_{ID_q,\theta,1} \) and \( KU_{ID_q,T,\theta,1} \). Note that even when \( r_{ID_q,\theta} \leftarrow R Z_p^{k} \) and \( t_{ID_q,T,\theta} \) are fixed, \( (SK_{ID_q,\theta,0} \cdot SK_{ID_q,\theta,2} \cdot \hat{S}K_{ID_q,\theta,\ell})_{\ell \in [1, |ID_q| + 1, L]} \) and \( (KU_{ID_q,T,\theta,0} \cdot KU_{ID_q,T,\theta,2}) \) do not reveal the quantities of \( r_{ID_q,\theta} \leftarrow R Z_p \) and \( \hat{t}_{ID_q,T,\theta} \leftarrow R Z_p \) since they are masked by \( \hat{r}_{ID_q,\theta} \leftarrow R Z_p^k \) and \( \hat{t}_{ID_q,T,\theta} \leftarrow R Z_p^{k} \) respectively. In \( Game_{1,6,q,1} \), for all nodes \( \theta \in BT_{ID_q} \) that correspond to \( ku_{ID_q,T,\theta} \) for \( T \geq T_{RL} \), \( SK_{ID_q,\theta,1} \) and \( KU_{ID_q,T,\theta,1} \) with the same nodes are distributed as follows:

- **SK_{ID_q,\theta,1}**:

  \[
  SK_{ID_q,\theta,1} = [k_{ID_q,\theta} + \hat{r}_{ID_q,\theta} a_q^{-1}]_2 \cdot [(V_0 + id_{L+1} V_1 + \cdots + id_{|ID_q|} V_{|ID_q|}) Zr_{ID_q,\theta}]_2,
  \]

- **KU_{ID_q,T,\theta,1} and KU_{ID_q,T,\theta,1}** for \( T < T_{RL} \):

  \[
  KU_{ID_q,T,\theta,1} = [k_{ID_q,\theta} + \hat{K}_{ID_q,T} - \hat{t}_{ID_q,T,\theta} a_q]_2^{-1} \cdot [(V_0 + TV_{L+1}) Zr_{ID_q,T,\theta}]_2,
  \]

  \[
  KU_{ID_q,T,\theta,1} = [k + a_q^{-1} + \hat{K}_{ID_q,T}]_2 \cdot [(V_0 + id_{q,1} V_1 + \cdots + id_{|ID_q|} V_{|ID_q|}) Zr_{ID_q,T,\theta}]_2
  \]

  \[
  \cdot [(V_0 + TV_{L+1}) Zr_{ID_q,T,\theta}]_2.
  \]

- **KU_{ID_q,T,\theta,1} and KU_{ID_q,T,\theta,1}** for \( T \geq T_{RL} \):

  \[
  KU_{ID_q,T,\theta,1} = [k_{ID_q,\theta} + \hat{K}_{ID_q,T}]_2^{-1} \cdot [(V_0 + TV_{L+1}) Zr_{ID_q,T,\theta}]_2
  \]

  \[
  KU_{ID_q,T,\theta,1} = [k + \hat{K}_{ID_q,T}]_2 \cdot [(V_0 + id_{q,1} V_1 + \cdots + id_{|ID_q|} V_{|ID_q|}) Zr_{ID_q,T,\theta}]_2
  \]

  \[
  \cdot [(V_0 + TV_{L+1}) Zr_{ID_q,T,\theta}]_2 \cdot [\hat{r}_{a_q^{-1}}]_2^{v_0+v_1 id_{q,1} + \cdots + v_{|ID_q|} id_{|ID_q|}}.
  \]
where \( \tilde{r}_{ID,\theta} \leftarrow R Z_p \), \( \tilde{t}_{ID,q,\theta} \leftarrow R Z_p \), \( \text{delk}_{ID,q,\theta} = k_{ID,q,\theta} \leftarrow R Z_p^{k+1} \), and \( \overline{\text{delk}}_{ID,q,T} = \overline{k}_{ID,q,T} \leftarrow R Z_p^{k+1} \). In contrast, the above distribution can be written as follows:

- \( SK_{ID,\theta,1} \):
  \[
  SK_{ID,\theta,1} = \left[ (k_{ID,\theta} + \alpha a^\perp) + (\tilde{r}_{ID,\theta} - 1) \alpha a^\perp \right]_2 \cdot \left[ (V_0 + id_{1}V_1 + \cdots + id_{|ID|}V_{|ID|}) \cdot Z_{R|ID,\theta} \right]_2,
  \]

- \( KU_{ID,q,T,\theta,1} \) and \( KU_{ID,q,T,\theta,1} \) for \( T < T_{RL} \):
  \[
  KU_{ID,q,T,\theta,1} = \left[ (k_{ID,q,\theta} + \alpha a^\perp) + \overline{k}_{ID,q,T} - (\tilde{r}_{ID,q,T,\theta} + 1) \alpha a^\perp \right]_2 \cdot \left[ (V_0 + TV_{L+1}) \cdot Z_{T_{ID,q,T,\theta}} \right]_2,
  \]

- \( KU_{ID,q,T,\theta,1} \) and \( KU_{ID,q,T,\theta,1} \) for \( T \geq T_{RL} \):
  \[
  KU_{ID,q,T,\theta,1} = \left[ (k + \alpha a^\perp + \overline{k}_{ID,q,T}) \cdot \left[ (V_0 + id_{q,1}V_1 + \cdots + id_{q,|ID|}V_{|ID|}) \cdot Z_{T_{ID,q,T}} \right]_2
  \]

where each \( \tilde{r}_{ID,\theta} - 1 \) and \( \tilde{t}_{ID,q,T,\theta} + 1 \) is distributed in \( Z_p \) uniformly at random, and each \( k_{ID,q,\theta} + \alpha a^\perp \) and \( \overline{k}_{ID,q,T} - \alpha a^\perp \) for \( T \geq T_{RL} \) is distributed in \( Z_p^{k+1} \) uniformly at random. Therefore, the above distribution is the same as the distribution in \( Game_{1,6.q,2} \) by setting \( \tilde{r}_{ID,\theta} - 1 \) and \( \tilde{t}_{ID,q,T,\theta} + 1 \) as the randomnesses in (10) and (23), respectively, and \( \text{delk}_{ID,q,\theta} = k_{ID,q,\theta} + \alpha a^\perp, \overline{\text{delk}}_{ID,q,T} = \overline{k}_{ID,q,T} - \alpha a^\perp \). We note that the claim holds for all \( T \geq T_{RL} \) and all nodes \( \theta \) that correspond to \( KU_{ID,q,T,\theta} \) for \( T \geq T_{RL} \), simultaneously. Thus, we complete the proof of Lemma 30.

**Lemma 31** (Sub-secret Key Transition from Pseudo-SF to Semi-functional, \( Game_{1,6.q,2} \approx_c Game_{1,6.q,3} \). \( Game_{1,6.q,2} \) and \( Game_{1,6.q,3} \) are computationally indistinguishable under the MDDH assumption in \( \mathbb{G}_2 \). Specifically, for any PPT Type-I adversary \( A \) making at most \( Q_{gen} \) secret key generation queries, there exists a reduction algorithm \( B_{1,6} \) such that

\[
|\text{Adv}_{1,6.q,2}(\lambda) - \text{Adv}_{1,6.q,3}(\lambda)| \leq \text{Adv}_{\mathbb{G}_2}^{\text{MDDH-}\mathbb{G}_2}(\lambda) + \frac{2}{p - 1}
\]

and \( T(B_{1,6}) \approx T(A) + Q_{gen}|T| \cdot \text{poly}(\lambda, L) \), where \( \text{poly}(\lambda, L) \) is independent of \( T(A) \).

We omit the detailed proof of Lemma 31 since it is almost the same as the proof of Lemma 13. By combining Lemmata 29–31, we have

\[
|\text{Adv}_{1,6}(\lambda) - \text{Adv}_{1,7}(\lambda)| \leq \sum_{q \in \{Q_{gen}\}} |\text{Adv}_{1,6.q-1,3}(\lambda) - \text{Adv}_{1,6.q,1}(\lambda)|
\]

\[
+ \sum_{q \in \{Q_{gen}\}} |\text{Adv}_{1,6.q,1}(\lambda) - \text{Adv}_{1,6.q,2}(\lambda)|
\]

\[
+ \sum_{q \in \{Q_{gen}\}} |\text{Adv}_{1,6.q,2}(\lambda) - \text{Adv}_{1,6.q,3}(\lambda)|
\]

\[
\leq Q_{gen} \cdot \sum_{j \in [2]} \text{Adv}_{\mathbb{G}_{1,7+4}}^{\text{MDDH-}\mathbb{G}_2}(\lambda) + \frac{4Q_{gen}}{p - 1}.
\]

Thus, we complete the proof of Lemma 28.  
\( \square \)
Lemma 32 (Final Transition, Game$_{1,7}$ $\equiv$ Game$_{1,8}$). Game$_{1,7}$ and Game$_{1,8}$ are identically distributed with probability $1 - 1/p$. Specifically, for any Type-I adversary $\mathcal{A}$, it holds that

$$|\text{Adv}_{\text{RHIBE}}(\lambda) - \text{Adv}_{\lambda}(\lambda)| = \frac{1}{p}.$$ 

We omit the proof of Lemma 32 since it is almost the same as the proof of Lemma 18.

By combining with Lemmata 3, 19, 20, 21, 22, 27, 28, and 32, against the Type-I adversary we have

$$\text{Adv}_{\text{RHIBE}}(\lambda) \leq |\text{Adv}_{\lambda}(\lambda) - \text{Adv}_{\lambda}(\lambda)| + \text{Adv}_{\lambda}(\lambda)$$

We obtain the inequality of Lemma 1. $\square$

7 Comparison

In this section, we compare our proposed RHIBE schemes with other known RHIBE schemes achieving the same property. We use the SXDH assumption for instantiating the schemes that are secure under the $k$-linear assumption. Columns $|\text{MPK}|$, $|\text{ct}_{\lambda}|$, $|\text{sk}_{\lambda}|$, $|\text{dk}_{\lambda}|$, and $|\text{ku}_{\lambda}(\lambda),\lambda|$ present a comparison of the size of MPK, ct$_{\lambda}$, sk$_{\lambda}$, dk$_{\lambda}$, and ku$_{\lambda}(\lambda),\lambda$ respectively. In the column #pairing, the number of pairing computations for the Dec algorithm are compared.
7.1 Comparison among RHIBE Schemes with Compact Ciphertexts

Table 7: Comparison of RHIBE schemes with compact ciphertexts

| Scheme | security | |MPK| |ct_{ID,T}| |dk_{ID,T}|
|--------|----------|-----------------|-----------------|-----------------|
| SE15 [SE15] | selective | (L + 6)|G| |3|G| + |G_T| |3|G|
| ETW20 [ETW20] | adaptive | (L + 5)|G_1| + 2|G_T| |3|G_1| + |Z_p| + |G_T| |5|G_2|
| Our Scheme | adaptive | (L + 5)|G_1| + 2|G_T| |4|G_1| + 2|Z_p| + |G_T| |8|G_2|

| Scheme | |sk_{ID}| |#pairing|
|--------|-----------------|-----------------|
| SE15 [SE15] | |PRF| + (L - |ID| + 2)|sk_{ID,θ}| |G| |3|
| ETW20 [ETW20] | |2(L - |ID|) + 7)(|delk_{ID,θ}| + |sk_{ID,θ}|)|G_2| |3|
| Our Scheme | |2|delk_{ID,θ}| |Z_p| + (2(L - |ID|) + 5)|sk_{ID,θ}| |G_2| |4|

| Scheme | |ku_{pa(ID),T}| |assump.| reduction loss|
|--------|-----------------|-----------------|-----------------|
| SE15 [SE15] | |(L - |pa(ID)| + 3)|ku_{pa(ID),T,θ}| |G| |q-type| O(L)|
| ETW20 [ETW20] | |(2(L - |pa(ID)|) + 5)|ku_{pa(ID),T,θ}| |G_2| |SXDH| O(LQ^2_{gen}|T|)|
| Our Scheme | |(2(L - |ID|) + 9 + 5|ku_{pa(ID),T,θ}|)|G_2| |SXDH| O(Q_{gen}(Q_{gen} + |T|))|

Table 7 compares our proposed RHIBE scheme with the other RHIBE schemes with compact ciphertexts [SE15, ETW20], i.e., Seo and Emura’s selectively secure scheme (SE15) and Emura et al.’s scheme (ETW20). Since we modify Chen and Gong’s HIBE scheme [CG17] for constructing the proposed RHIBE scheme, we use the same Chen and Gong’s HIBE scheme to instantiate Emura et al.’s semi-generic construction. We note that |delk_{ID,θ}|, |sk_{ID,θ}|, and |ku_{pa(ID),T,θ}| are the same among all the schemes except that SE15 does not depend on |delk_{ID,θ}|. All schemes have similar sizes of MPK, ct_{ID,T}, and dk_{ID,T} and almost the same |#pairing|. Although |sk_{ID}| is much larger than that of the selectively secure SE15, it is much shorter than that of the adaptively secure ETW20. We achieve the parameter saving since a delegation key delk_{ID,θ} of SE15 consists of 2(L - |ID|) + 7 G_2 elements, while that of ours consists of two Z_p elements. Moreover, |ku_{pa(ID),T}| of our scheme is much shorter than those of both SE15 and ETW20. We achieve the parameter saving due to the existence of helper key update ku_{pa(ID),T} that consists of 2(L - |ID|) + 9 G_2 elements. Specifically, sub-key updates ku_{pa(ID),T,θ} of SE15 and ETW20 consists of L - |pa(ID)| + 3 and 2(L - |pa(ID)|) + 5 G_2 elements, while that of ours consists of five G_2 elements. Although the security of SE15 is based on the non-standard q-type assumption, the security of ETW20 and ours are based on the same k-linear assumption. Unlike SE15 and ETW20, the reduction loss of our scheme does not depend on L, while that of SE15 is tighter than ours. We achieve strictly tighter reduction than ETW20.

7.2 Comparison among RHIBE Schemes with Adaptive Security
Table 8: Comparison of RHIBE schemes with adaptive security

| Scheme               | |MPK| |ct_{ID,T}| |sk_{ID}| |ku_{ID,T}| |dk_{ID,T}|
|---------------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|
| ETW20 [ETW20]       | O(L)            | O(1)            | O((L - |L\#delk_{ID,θ}| + O((L - |log λ|
| LK21 [LK21] (basic) | O(L)            | O(εL)          | O(L - ε)        | O(RεL + ε)      | O(ε)            |
| LK21 [LK21] (shorter ct) | O(L + λ)      | O(ε)            | O(L + λ - ε)    | O(RεL^2 + ε)    | O(ε)            |
| ETW21 [ETW21] (basic) | O(L)            | O(εL)          | O(L - ε)        | O(RεL + ε)      | O(ε)            |
| ETW21 [ETW21] (shorter ct) | O(L + M)      | O(εL/\lambda)| O(L + M - ε)    | O(RMεL + ε)     | O(ε)            |
| ETW21 [ETW21] (shorter ku) | O(L + M)      | O(εL/\lambda)| O(L + M - ε)    | O(RεL/\lambda + ε) | O(ε)            |
| Ours                | O(L)            | O(1)            | O(#delk_{ID,θ}) + O((L - |log λ|
|                     |                 |                 |                 |                 |                 |                 |

Table 8 compares the asymptotic space efficiency of adaptively secure RHIBE schemes [ETW20, LK21, ETW21], i.e., Emura et al.’s semi-generic construction (ETW20), Lee and Kim’s generic construction (LK21), and Emura et al.’s generic construction (ETW21). Since we modify Chen and Gong’s HIBE scheme [CG17] for constructing the proposed RHIBE scheme, we use the same Chen and Gong’s HIBE scheme to instantiate all the (semi-)generic constructions. We use a notation |ID| = ε for simplicity. Here, we assume that ETW20 and our scheme use binary trees with N = ε^L leaves as claimed in Section 3.1, while KL21 and ETW21 use binary trees with N = 2^{O(εL)} leaves since the latter use collision-resistant hash functions to assign every ID. Let R denote the number of users in RL_{ID,T}. As we claimed in Section 3.1, it holds that |KUN_{pa(ID),T}| = O(R log N/R). Here, we set |KUN_{pa(ID),T}T| ≈ O(R log λ) in the cases of ETW20 and our scheme, while we set |KUN_{pa(ID),T}| ≈ O(RλL) in the cases of LK21 and ETW21 for simplicity. The parameter M used in shorter ct variant of ETW21 is an integer such that 1 ≤ M ≤ λ, while M used in shorter ku variant of ETW21 is a non-negative integer.

Since we compare our scheme with ETW20 in Section 7.1, we here compare our scheme with LK21 and ETW21. At first, we compare our scheme with the basic schemes of LK21 and ETW21 that have the same asymptotic efficiency. All LK21, ETW21, and our scheme have the same size of MPK. The main bottleneck of our scheme is a large |sk_{ID}| that depends on #delk_{ID,θ} and log λ, while those of LK21 and ETW21 do not depend on #delk_{ID,θ} and log λ. In contrast, we achieve constant-size of |ct_{ID,T}| and |dk_{ID,T}|, while those of LK21 and ETW21 depend on εL and ε, respectively. Moreover, |ku_{ID,T}| of our scheme tends to be smaller than those of LK21 and ETW21 since we can use binary trees with less leaves than N than LK21 and ETW21.

Next, we compare our scheme with shorter ct variants of LK21 and ETW21. When we set M = Θ(λ), the shorter ct variants of LK21 and ETW21 have the same asymptotic efficiency. Although |sk_{ID}| of shorter ct variants of LK21 and ETW21 become larger than their basic schemes, they are still much shorter than that of our scheme. In contrast, all the other |MPK|, |ct_{ID,T}|, |ku_{ID,T}|, and |dk_{ID,T}| of our schemes are smaller than those of the shorter ct variants of LK21 and ETW21.
Finally, we compare our scheme with shorter $ku$ variant of ETW21. Although $|sk_{ID}|$ of the shorter $ku$ variant of ETW21 becomes larger than their basic schemes, they are still much shorter than that of our scheme. In contrast, $|MPK|$, $|ct_{ID,T}|$, and $|dk_{ID,T}|$ of our scheme are smaller than those of the shorter $ku$ variant of ETW21 regardless of the selections of parameter $M$. When we set a parameter $M = o(\lambda/\log \lambda)$, $|ku_{ID,T}|$ of our scheme is also smaller than that of the shorter $ku$ variant of ETW21. In other words, $|ku_{ID,T}|$ of the shorter $ku$ variant of ETW21 is smaller than that of our scheme only when $M = \Omega(\lambda/\log \lambda)$.

8 Conclusion

We propose an adaptively secure RHIBE scheme with compact ciphertexts under the standard $k$-linear assumption. The adaptive security of the previous scheme proposed by Emura et al. [ETW20] was proved by reducing the adaptive security of the underlying HIBE scheme to the adaptive security of their RHIBE scheme. In contrast, we proved the adaptive security of the proposed scheme directly by the dual system encryption methodology. Thus, we achieved a tighter reduction than that of Emura et al.’s scheme. Moreover, our scheme has much shorter secret keys and key updates than that of Emura et al. with ciphertexts made compact by a factor $O(L - |ID|)$. Since each parent user of the current adaptively secure RHIBE scheme has to store delegation keys whose number grows at least linearly with the number of children users, reducing the size of secret keys may pose a major problem by maintaining compact ciphertexts.

References


[CG17] Jie Chen and Junqing Gong. ABE with tag made easy - concise framework and new instantiations in prime-order groups. In Tsuyoshi Takagi and Thomas Peyrin, editors, Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the


