PLOS ONE

RESEARCH ARTICLE

Unbounded and revocable hierarchical identity-based encryption with adaptive security, decryption key exposure resistant, and short public parameters

Qianqian Xing[☯], **Baosheng Wang***, **Xiaofeng Wang**[☯], **Jing Tao**

College of Computer, National University of Defense Technology, Changsha, Hunan, China

☯ These authors contributed equally to this work.
* bswang@nudt.edu.cn

Citation: Xing Q, Wang B, Wang X, Tao J (2018) Unbounded and revocable hierarchical identity-based encryption with adaptive security, decryption key exposure resistant, and short public parameters. PLoS ONE 13(4): e0195204. https://doi.org/10.1371/journal.pone.0195204

Editor: Muhammad Khurram Khan, King Saud University, SAUDI ARABIA

Received: November 2, 2017

Accepted: March 19, 2018

Published: April 12, 2018

Data Availability Statement: All relevant data are within the paper. My manuscript focuses on a problem in the cryptography research. All the data for the comparison and evaluation is refered to the relative papers in the references of the paper.

Funding: This work was supported by National Basic Research and Development Program of China (973 Program), No. 2012CB315906, http://program.most.gov.cn/, Baosheng Wang (study design, and decision to publish is his role); and National Key Research and Development Program

Abstract

Revocation functionality and hierarchy key delegation are two necessary and crucial requirements to identity-based cryptosystems. Revocable hierarchical identity-based encryption (RHIBE) has attracted a lot of attention in recent years, many RHIBE schemes have been proposed but shown to be either insecure or bounded where they have to fix the maximum hierarchical depth of RHIBE at setup. In this paper, we propose a new unbounded RHIBE scheme with decryption key exposure resilience and with short public system parameters, and prove our RHIBE scheme to be adaptively secure. Our system model is scalable inherently to accommodate more levels of user adaptively with no adding workload or restarting the system. By carefully designing the hybrid games, we overcome the subtle obstacle in applying the dual system encryption methodology for the unbounded and revocable HIBE. To the best of our knowledge, this is the first construction of adaptively secure unbounded RHIBE scheme.

1 Introduction

Revocation functionality is indispensable to (H)IBE since there are threats of leaking a secret key by hacking or legal situation of expiration of contract for using system. In those seminal works [1] [2], it has also been pointed out that providing an efficient key hierarchy delegation mechanism for IBE is essential. To satisfing both hierarchical key delegation and user revocation, revocable hierarchical identity-based encryption (RHIBE) has been paid attention. Unfortunately most of existing RHIBEs proposed [1] [3] [4] [5] [6] [7] are either insecure or bounded where they have to fix the maximum hierarchical depth of RHIBE at setup. Bounded (R)HIBE schemes restrict the maximum hierarchy of (R)HIBE, i.e., they need to declare the max level in the public parameters at setup phase. It is highly impossible to set the maximum hierarchy properly in practice: too small to accommodate enough users or too large that wastes identity space needlessly and increase keys computation unnecessarily.

of China, No. 2017YFB0802301, http://program.most.gov.cn/, Xiaofeng Wang (data analysis and preparation of the manuscript is his role).

Competing interests: The authors have declared that no competing interests exist.

In contrast, the unbounded RHIBE is more scalable to achieve efficient and dynamic user management. Ryu proposed an unbounded RHIBE scheme [7] inspired by an universe KP-ABE [8]. But it only achieves selective-ID security. In selective-ID security notion, the reduction algorithm requires the challenge identity before the setup phase in the proof [1, 3]. That means the adversary holds no information before giving the challenge ID, but the simulator can exploit the challenge information submitted by the adversary to construct the trick public parameters and other keys in games. That is a weaker security notion.

Adaptive-ID security represents full security notion that an adversary gives the challenge identify when he has learnt the public information. Lee [5] considered the adaptively secure RHIBE but his scheme don't support the property of unbounded hierarchical key delegation. Xing [9] claimed to achive the first adaptively secure and unbounded RHIBE, but its security proof that uses the dual system encryption technique has some flaws. Therefore, the construction of an adaptively secure unbounded RHIBE scheme is still an unsolved open problem.

1.1 Our techniques

The dual system encryption framework [10] is usually for proving the adaptive security of HIBEs in composite-order bilinear groups. To achieve the adaptive security in the framework, the notion of semi-functionality is introduced [10] [11] and the proof strategy is that a normal challenge ciphertext is changed to be semi-functional, and then each normal private key is changed to be semi-functional one by one through hybrid games.

There is a paradox that need to be overcome. Since a normal ciphertext can be decrypted by a semi-functional private key but a semi-functional ciphertext cannot be decrypted by a semi-functional private key, a simulator can check whether a private key is normal or semi-functional by decrypting a semi-functional ciphertext(note that a simulator can generate a ciphertext and a private key for any identity). To overcome the obstacle, the nominally semi-functional type of private keys is introduced: the challenge semi-functional private key is constructed as a nominally semi-functional private key so that the semi-functional ciphertext of the same identity the simulator generates always can be decrypted by it. In addition, a detailed information theoretic argument should be given to argue that a nominally semi-functional key is indistinguishable from a semi-functional key.

Although the dual system encryption is maturing to exploit in normal HIBEs to achieve the adaptive security, it is more complex when dealing with revocable HIBE schemes. In HIBE, the essential restriction for the information theoretic argument is that an adversary cannot query a private key for ID that is a prefix of the challenge identity ID^*. However, the restriction do not exist in RHIBEs. The private key of any prefix of ID^* and the update key for the challenge time T^* are both allowed to query for the adversary in RHIBEs. Recall that the simulator of an HIBE scheme can change the normal-private key to a semi-functional private key by using a nominally semi-functional key and the constraint $ID \notin Prefix(ID^*)$ of the security model. The nominally semi-functional key is indistinguishable from a semi-functional key by an information theoretic argument using the constraint $ID \notin Prefix(ID^*)$. However, in the case of (U-)RHIBE, a simple method cannot change the normal-private key to the semi-functional private key since the adversary can query and achieve the private key for any $ID \in Prefix(ID^*)$.

Moreover, an unbounded RHIBE scheme has so low entropy context that it is hard to execute an information-theoretic argument, which is different with those bounded RHIBE schemes. So the dual system encryption method in Lee-RHIBE [5] does not work. Although Lewko and Waters [12] has proposed a nested dual system encryption approach to allow a sufficient information-theoretic argument in a very localized context for unbounded HIBEs, the trival applying to a revocable extention scheme is inappropriate to hold the paradox

information theoretic argument. Unfortunately Xing and Wang [9] have neglected this important change, so that the proof of their unbounded RHIBE scheme is non-rigorous with flaws. Obviously the attacker can distinguish between the oracles they design for the game hoppings in [9], which is not as they claimed in Lemma 4.

To circumvent the subtle obstacle and apply the dual system encryption methodology for our adaptively secure unbounded RHIBE with decryption key exposure resistance, our strategy is threehold:

(1) We use a modular design strategy like [13] and construct the private keys and update keys from smaller component keys. A private key consists of many HIBE private keys that are related to a path in a binary tree and an update key also consists of many IBE private keys that are related to a cover set in a binary tree. The HIBE and IBE private keys can be grouped together if they are related to the same node in a binary tree. So we change to deal with the transformation of component HIBE and IBE keys in the hybrid games instead of directly with the private keys and update keys of RHIBE which cannot be simply changed from normal keys to semi-functional keys.

(2) We design a nested dual system encryption for revocable and hierarchical IBE schemes with the concept of ephemeral semi-functionality for secret keys, update keys, decryption keys and ciphertexts. To demonstrate a hybrid process of games to chellenge keys and ciphertexts, we define several oracles to simulate the different forms of the component HIBE and IBE keys which construct the semi-functional or ephemeral semi-functional secret keys, update keys and decryption keys.

(3) For showing an information theoretic argument under RHIBE model successfully, we firstly classify the behavior of an adversary as two types under the restriction of the RHIBE security model. The Type-1 adversary is restricted to queries on the secret keys of any hierarchical identity satistying $ID|_k \notin Prefix(ID_l^*)$, so we carefully re-design a sequence of hybrid games to show several times of information theoretic arguments successfully for the secret keys and avoid a potential paradox for the update keys. The Type-2 adversary is restricted to queries on the update keys on the time $T \notin T^*$, so we carefully re-design the other sequence of hybrid games to show several times of information theoretic argument successfully for the update keys and avoid a potential paradox for the secret keys.

1.2 Our result

We propose the first adaptively secure unbounded RHIBE in composite-order bilinear groups under simple static assumptions. It removes the limitation of the maximum hierarchical depth in the encryption system and accommodate more levels of user adaptively without adding workload or restarting the system. Our RHIBE scheme also supports decryption key exposure resistance by the key-randomization method which meets the strong security notion for R(H) IBE [14].

Compared to existing RHIBE schemes, it is the first RHIBE to achieve simultaneously adaptive-ID security, decryption key exposure resistance and unbounded key delegation, as shown in Table 1. In Table 2, we discuss the comparison about the efficiency of key space and decryption computation, noted that l is the maximum level of the hierarchy, h is the level of a user in the hierarchy, N is the number of maximum users in each level, r is the number of revoked users, t_e is the cost for performing a bilinear pairing, $|G|$ and $|G_T|$ are the sizes of one element in G and G_T respectively. Our RHIBE scheme has the short and constant public parameter which is independent with the maximum level of the system hierarchy. Moreover, our RHIBE reduces the size of the update key from $O(hr\log(N/r))$to $O(h + r\log(N/r))$.

Table 1. Comparisons among RHIBE schemes.

Game	Unbounded delegation	Adaptive-ID security	DKE resist.	Assumption
Seo(2013) [1]	×	×	×	DBDH
Seo(2015) [3]	×	×	√	q-Type
Seo(2015) [6]	×	*	√	static
Ryu(2015) [7]	√	×	√	q-RW2
Lee(2016) [13]	×	×	√	q-Type
Lee(2016) [5]	×	√	√	static
Xing(2016) [9]	√	*	√	static
Our RHIBE	√	√	√	static

Note: Security marked with * has flaws in its proof.

https://doi.org/10.1371/journal.pone.0195204.t001

1.3 Related works

Efficient user revocation in RHIBE. An efficient tree-based key updating technique called the complete subtree (CS) method is a specific instance of the subset cover framework of Naor et al. [15]. In the scalable RIBEs using the CS method [16] [17] [14] [18] [19], every user holds a secret key composed of $logN$ subkeys, where N is the number of all users, and only one subkey of a non-revoked user can be used to generate a decryption key. If we directly extend this mechanism to RHIBE scheme, the second-level user need to prepare $(logN)^2$ subkeys since for every subkey of his parent he needs to generate $logN$ subkeys respectively, which results to $(logN)^l$ subkeys for an l-level user. Tsai et al. simply set the update key as another secret key in their RHIBE scheme [4]. Their construction is just as a trivial combination of two concurrent HIBE system, one for the derivation of secret keys and another for update keys. Lack of any efficient method of update and revocation, the size of the update key depends on the size of users linearly instead of logarithmically. Moreover, his approach require a new key center for update keys (called delegated revocation authority, DRA). That double deployment of key centers increases the system cost. Seo and Emura proposed a revocable HIBE scheme [1] with (l^2logN)-size secret keys for a user, where l is the maximum hierarchical level. This history preserving update method leads to a lengthy history information in an update key and requires the recursive definition of secret keys and update keys. Afterward Seo proposed a RHIBE with $(l \cdot logN)$-size secret keys for a user by a history-free update method. Recently, Lee and Park [13] proposed a new RHIBE scheme with shorter private keys and update keys by combining a new HIBE scheme that has short intermediate private keys and the CS scheme in a modular way, where the size of the secret key is $(logN)$ and the size of the update key is $(l + rlog(N/r))$. Another revocation method called the subset difference (SD) method [20] was utilized to

Table 2. Comparisons among RHIBE schemes.cont.

Game	Dec. cost	CT Size	PP size	SK size	UK Size
Seo(2013) [1]	$(h+2)t_e$	$O(l)$	$(l+4)\|G\|$	$O(l^2\log N)$	$O(r\log(N/r))$
Seo(2015) [3]	$3t_e$	$O(1)$	$(l+6)\|G\|$	$O(l\log N)$	$O(lr\log(N/r))$
Seo(2015) [6]	$3t_e$	$O(1)$	$(l+6)\|G\| + \|G_T\|$	$O(l\log N)$	$O(lr\log(N/r))$
Ryu(2015) [7]	$(2h+2)t_e$	$O(h)$	$7\|G\| + \|G_T\|$	$O(h\log N)$	$O(hr\log(N/r))$
Lee(2016) [13]	$(2h+3)t_e$	$O(l)$	$6\|G\| + \|G_T\|$	$O(\log N)$	$O(l + r\log(N/r))$
Lee(2016) [5]	$4t_e$	$O(l)$	$(l+5)\|G\| + \|G_T\|$	$O(l\log N)$	$O(l + r\log(N/r))$
Xing(2016) [9]	$(3h+2)t_e$	$O(h)$	$7\|G\| + \|G_T\|$	$O(h\log N)$	$O(hr\log(N/r))$
Our RHIBE	$(4h+4)t_e$	$O(h)$	$9\|G\| + \|G_T\|$	$O(h\log N)$	$O(h + r\log(N/r))$

https://doi.org/10.1371/journal.pone.0195204.t002

construct the RHIBE in [3] [13] [5]. Although this method has better performance in the transmission complexity, it has larger secret key size than the CS method.

Security model of R(H)IBE. Decryption key exposure resistance (DKER) has be considered by Seo [14], which discusses about the case where several decryption keys $dk_{I^*, T}$ for the target identity I^* are leaked to an adversary but the target decryption key dk_{I^*, T^*} is not exposed. Another attacks should be considered like insiders attack [3]. Since the hierarchical structure in RHIBE determines that every user as a low-level KGC hold the state information about his low-level children users, a stronger security model than RIBE should be considered where it allows an insider adversary to access at least their own state information. The key re-randomization method [3] is an operable way to resist this attack and also decryption key enclosure attack mentioned in [3, 14].

Adaptive security of R(H)IBE. By employing dual system encryption methodology [10, 11], the adaptive-ID security can be directly proved in (H)IBE. But the security model of revocable HIBE is different from general HIBEs, since the system of RHIBE just not allow the decryption key query of the challenge identity and its ancestor at the challenge time, but allows the secret key query of the challenge identity and its ancestor identity. Therefore, the dual system encryption of RHIBE is more complex than general dual system of HIBE. Those adaptive-ID secure RHIBEs [6] [5] employed the dual system encryptions which are applicable to bounded schemes. Their proof strategy cannot be employed to unbounded (R)HIBE schemes, cause the limited entropy available in the public parameters in unbounded schemes makes it difficult to construct the nominally semi-functional key without information-theoretic exposure. By applying the dual system encryption methodology in prime-order, Yohei [21] realizes an RIBE scheme with constant-size public parameter under static assumptions in prime-order groups.

2 Preliminaries

2.1 Revocable HIBE

Definition 1 *We define a RHIBE scheme* π = (*Setup*, *GenKey*, *DeriveKey*, *UpdateKey*, *Encrypt*, *Decrypt*, *Revoke*) *as following*:

1. ***Setup***(1^{λ}): *It takes a security parameter* λ, *and outputs a master public key PP, a master secret key MK, initial state* ST_0, *and an empty revocation list RL. Note that we don't require the maximum number of users in each level as an input parameter, unlike the defination by all the bounded RHIBEs.*

2. ***GenKey***($ID|_k$, $ST_{ID|_{k-1}}$, *PP*): *This algorithm takes as input* $ST_{ID|_{k-1}}$ *and an identity* $ID|_k$ *outputs the secret key* $SK_{ID|_k}$, *and updates* $ST_{ID|_{k-1}}$.

3. ***UpdateKey***(*T*, $RL_{ID|_{k-1}}$, $DK_{ID|_{k-1},T}$, $ST_{ID|_{k-1}}$, *PP*): *This algorithm takes as input the revocation list* $RL_{ID|_{k-1}}$, *state information* $ST_{ID|_{k-1}}$, *the decryption key* $DK_{ID|_{k-1},T}$, *and a time period T. Then, it outputs the update key* $UK_{ID|_{k-1},T}$.

4. ***DeriveKey***($SK_{ID|_k}$, $UK_{ID|_{k-1},T}$, *PP*): *This algorithm takes as input* $SK_{ID|_k}$ *of* $ID|_k$ *and* $UK_{ID|_{k-1},T}$, *and outputs the decryption key* $DK_{ID|_k,T}$ *of* $ID|_k$ *at time T if* $ID|_k$ *is not revoked at T by the parent, else outputs* $\perp$.

5. ***Encrypt***($ID|_l$, *T*, *M*, *PP*): *This algorithm takes as input a message M,* $ID|_l$ *and the current time T and outputs the ciphertext CT.*

6. ***Decrypt***($CT_{ID|_l, T}$, $DK_{ID'|_k, T'}$, *PP*): *This algorithm takes as input* $CT_{ID|_l, T}$ *and* $DK_{ID'|_k, T'}$, *and outputs the message if* $ID'|_k$ *is a prefix of* $ID|_l$ *and T* $T = T'$, *else outputs* $\perp$.

7. ***Revoke***$(RL_{ID|_{k-1}}, ST_{ID|_{k-1}}, ID|_k, T)$: *This algorithm takes as input* $ID|_k$ *and* T, *updates* $RL_{ID|_{k-1}}$ *managed by* $ID|_{k-1}$, *who is the parent user of* $ID|_k$, *by adding* $(ID|_k, T)$.

Definition 2 *We define an experiment under the adaptive-ID security against chosen plaintext attacks model in [5], as named "IND-RID-CPA" security.*

$$Exp_{\pi,\mathcal{A}}^{IND-RID-CPA}(\lambda):$$

$$(MK, PP, RL\varepsilon, ST\varepsilon) \leftarrow Setup(1^{\lambda});$$

$$(M_0^*, M_1^*, ID|_k^*, T^*, ST) \leftarrow \mathcal{A}^{\mathcal{O}}(Find, PP);$$

$$b \xleftarrow{R} \{0,1\}; CT^* \leftarrow Encrypt(PP, ID|_k^*, T^*, M_b^*);$$

$$b' \leftarrow \mathcal{A}^{\mathcal{O}}(Guess, CT^*, ST); Return\ 1\ if\ b' = b\ and\ 0\ otherwise.$$

In the above experiment, O is a set of oracles $\{SKGen_Q(\cdot), KeyUp_Q(\cdot,\cdot), Revoke_Q(\cdot,\cdot), DKGen_Q(\cdot,\cdot)\}$ defined as follows:

- $SKGen_Q(\cdot)$: For $ID|_k \in \mathcal{I}^k$, it returns $SK_{ID|_k}$ (by running $GenKey(ID|_k, ST_{ID|_{k-1}}, PP) \rightarrow SK_{ID|_k}$).
- $KeyUp_Q(\cdot,\cdot)$: For $T \in \mathcal{T}$ and $BT_{ID|_{k-1}}$, it returns $KU_{T,\,ID|_{k-1}}$ (by running UpdateKey$(T, RL_{ID|_{k-1}}, DK_{ID|_{k-1}}, ST_{ID|_{k-1}}, PP) \rightarrow KU_t$).
- $Revoke_Q(\cdot,\cdot)$: For $ID|_k \in \mathcal{I}^k$ and $T \in \mathcal{T}$, it returns the updated revocation list RL (by running Revoke$(RL_{ID|_{k-1}}, ST_{ID|_{k-1}}, ID|_k, T)$).
- $DKGen_Q(\cdot,\cdot)$: For $ID|_k \in \mathcal{I}^k$ and $T \in \mathcal{T}$, it returns $DK_{ID|_k,\,T}$ (by running DeriveKey$(SK_{ID|_k}, UK_{ID|_{k-1},\,T}, PP) \rightarrow DK_{ID|_k,\,T}$).

$\mathcal{A}$ is allowed to issue the above oracles with the following restrictions:

1. $Revoke_Q(\cdot,\cdot)$ can be queried on time T if $KeyUp_Q(\cdot)$ was queried on T.
2. $DKGen_Q(\cdot,\cdot)$ cannot be queried on time T before $KeyUp_Q(\cdot)$ was queried on T.
3. If $\mathcal{A}$ requested a private key query for ID_k^* that is a prefix of ID_l^* where $k \leq l$, then the identity ID_k^* or one of its ancestors should be revoked at some time T where $T \leq T^*$.
4. A cannot request a decryption key query for the challenge identity $ID^*|_l$ or its ancestors on the challenge time T^*.
5. $\mathcal{A}$ cannot request a revocation query for $ID|_k$ on time T if he already requested an update key query for $ID|_k$ in time T.
6. $\mathcal{A}$ must query to KeyUp$(\cdot,\cdot)$ and Revoke$(\cdot,\cdot)$ for same identity in increasing order of time.

The advantage of $\mathcal{A}$ is defined as $Adv_A^{RHIBE}(\lambda) = |Pr(b = b') - 0.5|$. We say that RHIBE is IND-RID-CPA secure if for all PPT adversary $\mathcal{A}$, his advantage $Adv_A^{RHIBE}(\lambda)$ is negligible in the security parameter λ.

2.2 Complexity assumptions

We generate $(n, G, G_T, e) \leftarrow \mathcal{G}$ where G and G_T be cyclic groups with order N and $p = p_1\,p_2\,p_3, p_1, p_2, p_3$ are distinct prime numbers, e: $G \times G \rightarrow G_T$ is an efficient, nondegenerate bilinear map. We denote the subgroup of G with order p_i as G_{p_i}. We define a function $Adv_{\mathcal{G},\mathcal{A}}(\lambda) = |Pr[\mathcal{A}(D, T_1)] - Pr[\mathcal{A}(D, T_2)]|$ for any PPT algorithm $\mathcal{A}$ and parameters D, T_1, T_2.

Assumption 1. Let $g \xleftarrow{R} G_{p_1}$, $D = (\mathcal{G}, g)$, $T_1 \xleftarrow{R} G_{p_1 p_2}$, $T_2 \xleftarrow{R} G_{p_1}$, we say that $\mathcal{G}$ satisfies Assumption 1 if $Adv_{\mathcal{G},\mathcal{A}}(\lambda)$ is a negligible function of λ for any PPT algorithm is $\mathcal{A}$.

Assumption 2. Let $g \xleftarrow{R} G_{p_1}, g_2, X_2, Y_2 \xleftarrow{R} G_{p_2}, g_3 \xleftarrow{R} G_{p_3}, \alpha, s \xleftarrow{R} Z_n$, T_1 be $e(g,g)^{\alpha s}$, $T_2 \xleftarrow{R} G_T$, $D = (\mathcal{G}, g, g_2, g_3, g^{\alpha} X_2, g^{s} Y_2)$, we say that $\mathcal{G}$ satisfies Assumption 2 if $Adv_{\mathcal{G},\mathcal{A}}(\lambda)$ is a negligible function of λ for any PPT algorithm is $\mathcal{A}$.

Assumption 3. Let $g, X_1 \xleftarrow{R} G_{p_1}, g_2 \xleftarrow{R} G_{p_2}, X_3 \xleftarrow{R} G_{p_3}, T_1 \xleftarrow{R} G_{p_1}, T_2 \xleftarrow{R} G_{p_1 p_3}$, $D = (\mathcal{G}, g, g_2, X_1 X_3)$, we say that $\mathcal{G}$ satisfies Assumption 3 if $Adv_{\mathcal{G},\mathcal{A}}(\lambda)$ is a negligible function of λ for any PPT algorithm is $\mathcal{A}$.

Assumption 4. Let $g, X_1 \xleftarrow{R} G_{p_1}, X_2, Y_2 \xleftarrow{R} G_{p_2}, g_3, Y_3 \xleftarrow{R} G_{p_3}, T_1 \xleftarrow{R} G_{p_1 p_3}, T_2 \xleftarrow{R} G$, $D = (\mathcal{G}, g, g_3, X_1 X_2, Y_2 Y_3)$, we say that $\mathcal{G}$ satisfies Assumption 4 if $Adv_{\mathcal{G},\mathcal{A}}(\lambda)$ is a negligible function of λ for any PPT algorithm is $\mathcal{A}$.

3 Design of U-RHIBE system

We firstly describe the key encapsulation mechanism (KEM) version of the unbounded HIBE scheme [12] and its 1-level (H)IBE scheme that are used as the building blocks of our RHIBE schemes. Let $GS = ((N = p_1 p_2 p_3, G, G_T, e), g, g_2, g_3) \leftarrow \mathcal{G}(\lambda)$ be the bilinear group, where λ is a security parameter and g_2 denotes a generator of G_{p_2}, g_3 denotes a generator of G_{p_3} and g be a generator of G_{p_1}.

3.1 HIBE scheme

We define a key-group function $\kappa(I, y, r)$ as the group elements

$$\kappa(I, y, r) = (w^{y}, g^{y}, g^{r}, (u^{I} h)^{r} v^{y})$$

and an expression $g^{\lambda}\,\kappa(I, y, r)$ as

$$g^{\lambda}\kappa(I, y, r) = (g^{\lambda} w^{y}, g^{y}, g^{r}, (u^{I} h)^{r} v^{y})$$

HIBE.Setup(GS): It selects $u, h, w, v \xleftarrow{R} G_{p_1}$ and $\alpha \xleftarrow{R} Z_p$. It outputs a master key $MK = \alpha$ and public parameters $PP = ((p, G, G_T, e), g, u, h, w, v, \Omega = e(g,g)^{\alpha})$.

HIBE.GenKey($ID|_k$, MK, PP): Let the identity $ID|_k = (I_1, \ldots, I_k) \in \mathcal{I}^k$, and $\mathcal{I} = \{0,1\}^{\lambda}$ be the identity space. It chooses $\lambda_1 \cdots \lambda_k, r_1 \cdots r_k, y_1 \cdots y_k \xleftarrow{R} Z_p$ where $\lambda_1 + \cdots + \lambda_k = \alpha$ and outputs a private key $SK_{ID|_k} = \{K_i = g^{\lambda_i} \kappa(I_i, y_i, r_i), i = 1 \cdots k\}$.

HIBE.RandKey($ID|_k$, $SK_{ID|_k}$, PP): Let $SK_{ID|_k} = (\{K'_{i,0}, K'_{i,1}, K'_{i,2}, K'_{i,3}\}_{i=1}^{k})$. It chooses $\lambda_1 \cdots \lambda_k, r_1 \cdots r_k, y_1 \cdots y_k \xleftarrow{R} Z_p$ where $\lambda_1 + \cdots + \lambda_k = 0$ and outputs a re-randomized private key $SK_{ID|_k} = \{K'_{i,0} g^{\lambda_i} w^{y_i}, K'_{i,1} g^{y_i}, K'_{i,2} g^{r_i}, (u^{I_i} h)^{r_i} v^{y_i} \cdot K'_{i,3}\}_{i=1}^{k})$.

HIBE.Delegate($ID|_k$, $SK_{ID|_{k-1}}$, PP): Let $SK_{ID|_{k-1}} = (\{K'_{i,0}, K'_{i,1}, K'_{i,2}, K'_{i,3}\}_{i=1}^{k-1})$. It chooses $\lambda_1 \cdots \lambda_k, y_k, r_k \xleftarrow{R} Z_p$ where $\lambda_1 + \cdots + \lambda_k = 0$ and creates a temporal delegated private key $TSK_{ID|_k} = (\{K'_{i,0} g^{\lambda_i}, K'_{i,1}, K'_{i,2}, K'_{i,3}\}_{i=1}^{k-1}, g^{\lambda_k} \kappa(I_k, y_k, r_k))$. Next, it outputs a delegated private key $SK_{ID|_k}$ by running HIBE.RandKey($ID|_k$, $TSK_{ID|_k}$, PP).

HIBE.Encaps($ID|_l$, s, PP): Let $ID|_l = (I_1, \ldots, I_l) \in I^l$. It chooses $t_1, \cdots, t_k \xleftarrow{R} Z_p$ and outputs a ciphertext $CT_{ID|_l} = (g^{s}, \{w^{s} v^{t_i}, (u^{I_i} h)^{t_i}, g^{t_i}\}_{i=1}^{l})$ and a session key $EK = \Omega^{s}$.

HIBE.Decaps($CT_{ID|_l}$, $SK_{ID'|_k}$, PP): Let $CT_{ID|_l} = (C_0, \{C_{i,1}, C_{i,2}, C_{i,3}\}_{i=1}^{l})$, $SK_{ID'|_k} = (K_0, \{K_{i,1}, K_{i,2}, K_{i,3}\}_{i=1}^{k})$. If $ID'|_k$ is a prefix of $ID|_l$, it outputs a session key $EK = \prod_{i=1}^{k} (e(C_0, K_{i,0}) e(C_{i,3}, K_{i,3}) / (e(C_{i,1}, K_{i,1})\, e(C_{i,2}, K_{i,2})))$. Otherwise, it outputs $\perp$.

Additionally, we introduce two algorithms for our modular RHIBE construction, the ChangeKey algorithm and the MergeKey algorithm, which are defined similarly with the algorithms in [5].

HIBE.ChangeKey($SK_{ID|_K}$, δ, PP): Let $SK_{ID|_k} = (\{K'_{i,0}, K'_{i,1}, K'_{i,2}, K'_{i,3}\}_{i=1}^{k})$. It chooses $\lambda_1 \cdots \lambda_k \xleftarrow{R} Z_p$ where $\lambda_1 + \cdots + \lambda_k = \delta$ and sets $TSK = (\{K_{i,0} g^{\lambda_i}, K_{i,1}, K_{i,2}, K_{i,3}\}_{j=1}^{k})$. It outputs a new private key $SK_{ID|_k} \leftarrow HIBE.RandKey(ID|_k, TSK_{(n)}, PP)$.

HIBE.MergeKey($SK^{(1)}_{ID|_K}, SK^{(2)}_{ID|_K}, \eta, PP$): Let $SK^{(1)}_{ID|_k} = (\{K'_{i,0}, K'_{i,1}, K'_{i,2}, K'_{i,3}\}_{i=1}^{k})$ and $SK^{(2)}_{ID|_k} = (\{K''_{i,0}, K''_{i,1}, K''_{i,2}, K''_{i,3}\}_{i=1}^{k})$ be two private keys for the same identity $ID|_K$. It computes a temporal private key $TSK = (\{K'_{i,0} \cdot K''_{i,0}, K'_{i,1} \cdot K''_{i,1}, K'_{i,2} \cdot K''_{i,2}, K'_{i,3} \cdot K''_{i,3}\}_{i=1}^{k})$. Next, it outputs a merged private key $SK_{ID|_K} \leftarrow HIBE.ChangeKey(TSK, \eta, PP)$. Note that the master key part is $\alpha_1 + \alpha_2 + \eta$ if the master key parts of $SK^{(1)}_{ID|_K}$ and $SK^{(2)}_{ID|_K}$ are α_1 and α_2 respectively.

3.2 IBE scheme

A trivial extension to RHIBE from the HIBE in [12] constructs the decryption key of $(T, ID|_k)$ as $\{D_0 = g^{\lambda_0} \kappa(T, y_0, r_0), D_i = g^{\lambda_i} \kappa(I_i, y_i, r_i), i = 1 \cdots k | \sum_{i=0}^{k} \lambda_i = \alpha\}$. It remains some problem in the proof of RHIBE model, where the information theoretic argument is not easy to show as of the model of HIBE. So we modify the construction by defining a new update-key-group function as

$$\kappa_T(T, y, r) = (w_0^y, g^y, g^r, (u_0^T h_0)^r v_0^y) \tag{1}$$

and $D_0 = g^{\lambda_0} \kappa_T(T, y_0, r_0)$, which is constructed from the component IBE secret key.

IBE.Setup(GS): It selects $u, h, w, v \xleftarrow{R} G_{p_1}$ and $\alpha \xleftarrow{R} Z_p$. It outputs a master key $MK = \beta$ and public parameters $PP = ((p, G, G_T, e), g, u_0, h_0, w_0, v_0, \Omega = e(g, g)^{\beta})$.

IBE.GenKey(T, MK, PP): This algorithm takes as input a time T and the master key MK, and the public parameters PP. It chooses $r, y \xleftarrow{R} Z_p$ and outputs a IBE secret key $SK_T = g^{\alpha} \kappa_T(T, y, r)$.

IBE.RandKey(T, SK_T, PP): Let the private key be $SK_T = (K'_0, K'_1, K'_2, K'_3)$. It chooses $r, y \xleftarrow{R} Z_p$ and outputs a re-randomized private key $SK_T = (K_0 = K'_0 w_0^y, K_1 = K'_1 g^y, K_2 = K'_2 g^r, K_3 = K'_3 (u_0^T h_0)^r v_0^y)$.

IBE.Encaps(T, s, PP): It chooses $t \xleftarrow{R} Z_p$ and outputs a ciphertext $CT_T = (C_0 = g^s, C_1 = w_0^s v_0^t, C_2 = (u_0^T h_0)^t, C_3 = g^t)$ and the session key $EK = \Omega^s$.

IBE.Decaps(CT_T, $SK_{T'}$, PP): Let the ciphertext $CT_T = (C_0, C_1, C_2, C_3)$, the private key $SK_T = (K_0, K_1, K_2, K_3)$. If $T = T'$, it outputs a session key $EK = e(C_0, K_0) e(C_3, K_3) / (e(C_1, K_1) e(C_2, K_2))$. Otherwise, it outputs $\perp$.

The contruction of IBE.ChangeKey and IBE.MergeKey is similar with HIBE.ChangeKey and HIBE.MergeKey and we omit them here.

3.3 The CS method

We exploit the complete subtree (CS) method to construct our RHIBE scheme. We follow the definition of the CS scheme in the work of Lee and Park [22].

CS.Setup(N_{max}): Let $N_{max} = 2^n$. It first sets a full binary tree $\mathcal{BT}$ of depth n. Each user is assigned to a different leaf node in $\mathcal{BT}$. The collection S is defined as $\{S_i\}$ where S_i is the set of all leaves in a subtree $\mathcal{T}_i$ with a subroot $v_i \in \mathcal{BT}$. It outputs the full binary tree $\mathcal{BT}$.

CS.Assign($\mathcal{BT}$, ID): Let v_{ID} be a leaf node of $\mathcal{BT}$ that is assigned to the user ID. Let $(v_{k_0}, v_{k_1}, \cdots, v_{k_n})$ be the path from the root node $v_{k_0} = v_0$ to the leaf node $v_{k_n} = v_{ID}$. For all $j \in \{k_0, \cdots, k_n\}$, it adds S_j into PV_{ID}. It outputs the private set $PV_{ID} = \{S_j\}$.

CS.Cover($\mathcal{BT}$, R): It first computes the Steiner tree $ST(R)$. Let $\mathcal{T}_{k_1}, \cdots, \mathcal{T}_{k_m}$ be all the subtrees of $\mathcal{BT}$ that hang off $ST(R)$, that is all subtrees whose roots $v_{k_1}, \cdots, v_{k_m}$ are not in $ST(R)$ but adjacent to nodes of outdegree 1 in $ST(R)$. For all $i \in \{k_1, \cdots, k_m\}$, it adds S_i into CV_R. It outputs a covering set $CV_R = \{S_i\}$.

CS.Match(CV_R, PV_{ID}): It finds a subset S_k with $S_k \in CV_R$ and $S_k \in PV_{ID}$. If there is such a subset, it outputs S_k. Otherwise, it outputs $\perp$.

3.4 Construction

RHIBE.Setup(1^λ, N_{max}): The Setup algorithm takes a security parameter λ and a maximum number of users for each level N_{max} as input. It firstly runs $\mathcal{G}$ to obtains two groups G, G_T of order $p = p_1p_2p_3$, where p_1, p_2, p_3 are distinct primes, and a bilinear map e: $G \times G \rightarrow G_T$. It sets $GS = ((N, G, G_T, e), g, g_2, g_3)$ where g, g_2 and g_3 denote the generators of G_{p_1}, G_{p_2}, and G_{p_3} in order. It selects a random exponent $\alpha \in Z_p$, set Ω be $e(g, g)^\alpha$. It outputs a master key $MK = \alpha$ and public parameters $PP = (PP_{HIBE}, PP_{IBE}, \Omega, N_{max})$, where $PP_{HIBE} \leftarrow HIBE.Setup(GS)$, and $PP_{IBE} \leftarrow IBE.Setup(GS)$.

RHIBE.GenKey($ID|_k$, $ST_{ID|_{k-1}}$, PP): This algorithm takes as input an identity $ID|_k = (I_1, \ldots, I_k) \in \mathcal{I}^k$, the state $ST_{ID|_{k-1}}$ which contains $BT_{ID|_{k-1}}$.

1. If $ST_{ID|_{k-1}}$ is empty, it obtains $BT_{ID|_{k-1}} \leftarrow CS.Setup(N_{max})$ and then it sets $ST_{ID|_{k-1}} = (BT_{ID|_{k-1}}, \beta_{ID_{k-1}}, z_{ID_{k-1}})$, where $\beta_{ID_{k-1}}$ is a false master key and $z_{ID_{k-1}}$ is a PRF key.
2. It first assigns $ID|_k$ to a random leaf node $v \in BT_{ID|_{k-1}}$ and obtains a node set $Path(ID|_k) \leftarrow CS.Assign(BT_{ID|_{k-1}}, ID|_k)$ for $ID|_k$. For each $S_\theta \in Path$, it computes $\gamma_\theta = \mathbf{PRF}(z_{ID_{k-1}}, L_\theta)$ where $L_\theta = \mathbf{Label}(S_\theta)$ and obtains an HIBE private key $SK_{HIBE,S_\theta} \leftarrow HIBE.GenKey(ID|_k, \gamma_\theta, PP)$. Finally, it outputs a private key $SK_{ID|_k} = (Path, \{SK_{HIBE,S_\theta}\}_{S_\theta \in Path})$. Note that the master key part of SK_{HIBE,S_θ} is γ_θ.

RHIBE.UpdateKey(T, $RL_{ID|_{k-1}}$, $DK_{ID|_{k-1}}$, $ST_{ID|_{k-1}}$, PP): let $DK_{ID|_{k-1},T} = (RSK_{HIBE,ID|_{k-1}}, RSK'_{IBE,T})$, the state $ST_{ID|_{k-1}} = (BT_{ID|_{k-1}}, \beta_{ID|_{k-1}}, z_{ID|_{k-1}})$ with $k \geq 1$.

1. It first obtains a randomized decryption key $RDK_{ID|_{k-1},T}$ as $(RSK_{IBE}, RSK_{HIBE}) \leftarrow RHIBE.RandDK(DK_{ID|_k,T}, -\beta_{ID|_{k-1}}, PP)$.
2. It derives the set of revoked identities R at time T from $RL_{ID|_{k-1}}$. Next, it obtains a covering set $CV_R = \{S_i\}$ by running **CS.Cover**($BT_{ID|_{k-1}}$, R).
3. For each $S_i \in CV_R$, it computes $\gamma_i = \mathbf{PRF}(z_{ID_{k-1}}, L_i)$ where $L_i = \mathbf{Label}(S_i)$ and obtains an IBE private key $SK'_{IBE,S_i} \leftarrow IBE.GenKey(T, -\gamma_i, PP)$. Then It computes $SK_{IBE,S_i} \leftarrow IBE.MergeKey(SK'_{IBE,S_i}, RSK_{IBE}, \beta_{ID_{k-1}}, PP)$
4. It finally outputs an update key $UK_{ID|_{k-1},T} = (CV_R, \{SK_{IBE,S_i}, RSK_{HIBE}\}_{S_i \in CVR})$. Note that the master key parts of RSK_{HIBE} and SK_{IBE,S_i} are η' and $\alpha - \eta' - \gamma_i$ for some random η' respectively.

RHIBE.DeriveKey($SK_{ID|_k}$, $UK_{ID|_{k-1},T}$, PP): This algorithm takes as input a private key $SK_{ID|_k} = (Path, \{SK_{HIBE,S_\theta}\}_{S_\theta \in Path})$ for an identity $ID|_k$, an update key $UK_{ID|_{k-1},T} = (CV_R, \{SK_{IBE,S_i}, RSK'_{HIBE,ID|_{k-1}}\}_{S_i \in CV_R})$ for time T.

1. If $K = 0$, then $SK_{ID|_0} = MK = \alpha$ and $UK_{ID|_{-1},T}$ is empty. It selects a random exponent $\eta \in Z_p$. It then obtains $RSK_{HIBE,ID|_0} \leftarrow HIBE.GenKey(ID|_0, \eta, PP)$ and $RSK_{IBE,T} \leftarrow IBE.GenKey(T, \alpha - \eta, PP)$. It outputs a decryption key $DK_{ID|_0,T} = (RSK_{IBE,T}, RSK_{HIBE,ID|_0})$.

2. If $k \geq 1$, then if $ID|_k \notin RL_{ID|_{k-1}}$, then it obtains (S_i, S_i) by running **CS.Match**$(CV_R, Path)$. Otherwise, it outputs $\perp$. It derives SK_{HIBE,S_i} from $SK_{ID|_k}$ and SK_{IBE,S_i} from $UK_{ID|_{k-1},T}$.
3. It obtains $RSK''_{HIBE,ID|_k} \leftarrow HIBE.Delegate(ID|_k, RSK'_{HIBE,ID|_{k-1}}, PP)$ since $ID|_{k-1} \in \text{Prefix}(ID|_k)$. Next, it selects a random exponent $\eta \in Z_p$, obtains $RSK_{HIBE,ID|_k} \leftarrow HIBE.MergeKey(RSK''_{HIBE,ID|_k}, SK_{HIBE,S_i}, \eta, PP)$ and obtains $RSK_{IBE,T} \leftarrow IBE.ChangeKey(SK_{IBE,S_i}, -\eta, PP)$ respectively. Finally, it outputs a decryption key $DK_{ID|_k,T} = (RSK_{IBE,T}, RSK_{HIBE,\ ID|_k})$.

Note that the master key parts of $RSK_{HIBE,\ ID|_k}$ and $RSK_{IBE,\ T}$ are η' and $\alpha - \eta'$ for some random η' respectively.

RHIBE.RandDK$(DK'_{ID|_k,T}, \beta, PP)$: Let $DK'_{ID|_k,T} = (RSK'_{IBE,T}, RSK'_{HIBE,ID|_k})$, and $\beta \in Z_p$ be an exponent. It first selects a random exponent η and obtains $RSK_{HIBE,ID|_k} \leftarrow HIBE.ChangeKey(RSK'_{HIBE,ID|_k}, \eta, PP)$ and $RSK_{IBE,T} \leftarrow IBE.ChangeKey(RSK'_{IBE,T}, -\eta + \beta, PP)$. It outputs a re-randomized decryption key $DK_{ID|_k,T} = (RSK_{IBE,T}, RSK_{HIBE,\ ID|_k})$.

RHIBE.Encrypt$(ID|_l, T, M, PP)$: This algorithm takes as input an identity $ID|_l = (I_1, \ldots, I_l) \in \mathcal{I}^l$, time T, a message $M \in \mathcal{M}$. It chooses a random exponent $t \in Z_p$. Next it obtains $(CH_{HIBE,ID|_l}, EK_{HIBE}) \leftarrow HIBE.Encaps(ID|_l, t, PP)$. It also obtains $(CH_{IBE,T}, EK_{IBE}) \leftarrow IBE.Encaps(T, t, PP)$. It outputs a ciphertext $CT_{ID|_k,T} = (CH_{IBE,T}, CH_{HIBE,ID|_l}, C = \Omega^t \cdot M)$.

RHIBE.Decrypt$(CT_{ID|_l,T}, DK_{ID'|_k,T'}, PP)$: This algorithm takes as input a ciphertext $CT_{ID|_l,T} = (CH_{IBE,T}, CH_{HIBE,ID|_l}, C)$, a decryption key $DK_{ID'|_k,T'} = (RSK_{IBE,T'}, RSK_{HIBE,ID'|_k})$. If $ID'|_k$ is a prefix of $ID|_l$ and $T = T'$, then it obtains $EK_{HIBE} \leftarrow HIBE.Decaps(CH_{HIBE,ID|_l}, RSK_{HIBE,ID|_k}, PP)$ and $EK_{IBE} \leftarrow IBE.Decaps(CH_{IBE,T}, RSK_{IBE,T}, PP)$. Otherwise, it outputs $\perp$. It outputs an encrypted message by computing $M = C \cdot (EK_{HIBE} \cdot EK_{IBE})^{-1}$.

RHIBE.Revoke$(ID|_k, T, RL_{ID|_{k-1}}, ST_{ID|_{k-1}})$: This algorithm takes as input an identity $ID|_k$, revocation time T, the revocation list $RL_{ID|_{k-1}}$, and the state $ST_{ID|_{k-1}}$. If $(ID|_k, -) \notin ST_{ID|_{k-1}}$, then it outputs $\perp$ since the private key of $ID|_k$ was not generated. Otherwise, it adds $(ID|_k, T)$ to $RL_{ID|_{k-1}}$ and outputs the updated revocation list $RL_{ID|_{k-1}}$.

3.5 Correctness

If a user is not revoked at time T, the RHIBE.DeriveKey algorithm correctly derive his decryption key $DK_{ID|_k,T}$ as

$$(g^{\alpha - \Pi_{i=1}^{k} \lambda_i} w_0^{b_0}, g^{b_0}, g^{r_0}, (u_0^T h_0)^{r_0} v_0^{b_0}, \{g^{\lambda_i} w^{b_i}, g^{b_i}, g^{r_i}, (u^{I_i} h)^{r_i} v^{b_i}\}_{i=1}^{k})$$

The RHIBE.Decrypt algorithm takes $CT_{ID|_l,T}$ as input, where

$$CT_{ID|_l,T} = (g^s, w_0^s v_0^t, (u_0^T h_0)^t, g^t, \{w^s v^{t_i}, (u^{I_i} h)^{t_i}, g^{t_i}\}_{i=1}^{l}),$$

and computes $B = C/M$ as

$$B = \frac{e(g^s, g^{\alpha} w_0^{b_0} \prod_{i=1}^{k} w^{b_i}) e(g^{t_0}, (u_0^{I_0} h_0)^{r_0} v_0^{b_0})}{e(w_0^s v_0^{t_0}, g^{b_0}) e((u_0^T h_0)^{t_0}, g^{r_0})} \prod_{i=1}^{k} \frac{e(g^{t_i}, (u^{I_i} h)^{r_i} v^{b_i})}{(e(w^s v^{t_i}, g^{b_i}) e((u^{I_i} h)^{t_i}, g^{r_i}))} = \Omega^s$$

4 Security analysis

We use the dual system encryption proof techinique to prove the adaptive security of our U-RHIBE. We adopt the concept of *ephemeral semi-functionality* [12] and design a new nested dual system encryption for unbounded RHIBEs. As an intermediary transforming stage between the normal and semi-functional distributions, the ephemeral semi-functionality helps us to overcome the challenge presented by low entropy in the public parameters.

Theorem 1 *Our unbounded RHIBE scheme is IND-RID-CPA secure if Assumption 1–4 hold.*

Proof We firstly define the semi-functional type and the ephemeral semi-functional types of keys and ciphertexts in Sec.4.1 which represent the types of keys and ciphertexts answered to the queries in the challenge game. Secondly we conduct the security proof by the indistinguishabilities of a sequence of hybrid games that we define in Sec.4.2.

4.1 Definition of (ephemeral) semi-functional keys and ciphertexts

For constructing the different types of ciphertexts, secret keys, update keys and decryption keys, the challenger $\mathcal{B}$ is initially given renadom elements g, u, v, w, u_0, v_0, $w_0 \in G_{p_1}$, $g_2 \in G_{p_2}$, $g_3 \in G_{p_3}$, as well as random exponents ψ_1, ψ_2, σ_1, σ_2, a', b', s, δ_1, δ_2, γ.

We define the semi-functional ciphertext and five types of ephemeral semi-functional ciphertexts of a normal ciphertext $CT_{ID|_b T}$ by changing the C_0 element into $G_{p_1 p 2}$ and the $l+1$ numbers of the ciphertext-element-groups $(C_{i,1}, C_{i,2}, C_{i,3})$ into different types. The definations of ephemeral semi-functional ciphertexts called **ESF-1-CT**, **ESF-2$^{\mathbf{k}}$-CT**, **ESF-3$^{\mathbf{k}}$-CT**, **ESF-4$^{\mathbf{k}}$-CT** and **ESF-5-CT** where $0 \leq k \leq l$ are in Appendix.A. In the definations of the semi-functional ciphertext, we add G_{p_2} term on the first element of all ciphertext-element-groups.

RHIBE.EncryptSF: It firstly obtains the normal ciphertext $CT_{ID|_b T} = (C, C_0, \{C_{i,1}, C_{i,2}, C_{i,3}\}_{i=0}^{l})$ for an identity $ID|_l = (I_1, \ldots, I_k) \in \mathcal{I}^k$, a time $T \in \mathcal{T}$ and a message $M \in \mathcal{M}$. It chooses exponents $\gamma, \delta_1, \delta_2 \in \mathbf{Z}_p$ and outputs the SF-CT $\widetilde{CT}_{ID|_l,T}$ as

$$(C, C_0 \cdot g_2^{\gamma}, C_{0,1} \cdot g_2^{\delta_2}, C_{0,2}, C_{0,3}, \{C_{i,1} \cdot g_2^{\delta_1}, C_{i,2}, C_{i,3}\}_{i=1}^{l})$$

As we mentioned before, our normal secret key and update key cannot be simply changed to semi-functional keys as same as in [11] one by one owing to the inefficiency of the information theoretic argument in our scheme. And we divide secret keys and update keys into samll component keys which are group together if they are related to the same node in a binary tree.

We only change the last element-group of our normal secret key for constructing the semi-functional secret key and the ephemeral semi-functional secret key like in [11]. We define one type of semi-functional secret key and five types of ephemeral semi-functional secret key. The defination of ephemeral semi-functional secret key called **ESF-1-SK**, **ESF-2-SK**, **ESF-3-SK**, **ESF-4-SK** and **ESF-5-SK** are in Appendix.A. In the defination of the semi-functional secret key, we add $G_{p_2 p3}$ term on the first 2 elements and the last element of the last element-group.

RHIBE.SKeySF $(ID|_j, ST_{ID|_{j-1}}, PP, \theta) \rightarrow \widetilde{SK}_{HIBE,S_\theta}$: It constructs the correlative sub-key $SK_{HIBE,S_\theta} = (\{K_{i,0}, K_{i,1}, K_{i,2}, K_{i,3}\}_{i=1}^{j})$ to the node $\theta \in Path(ID_j)$ in the $BT_{ID|_{j-1}}$ as follows: It chooses random exponents $y', r \in \mathbf{Z}_p$ and choose $\sigma_1, \psi_1 \in \mathbf{Z}_p$, then it constructs $\kappa^{sf}(I_j, y', r)$ for the last element-group as

$$(w^{y'}(g_2 g_3)^{y'\psi_1}, g^{y'}(g_2 g_3)^{y'}, g^{r}, v^{y'}(g_2 g_3)^{y'\sigma_1}(u^{I_j} h)^{r})$$

And the contruction of the other element-groups follows the construction of SK_{HIBE,S_θ} in RHIBE.GenKey.

We define one type of semi-functional update key and five types of ephemeral semi-functional update key. The defination of ephemeral semi-functional update key called **ESF-1-UK**, **ESF-2-UK**, **ESF-3-UK**, **ESF-4-UK** and **ESF-5-UK** are in Appendix.A. The constructions from the normal component update key to the (ephemeral) semi-functional component update keys are similar to that of secret keys, expect that we change the first element group of normal component update key to different types.

RHIBE.UpdateKeySF $(T, ST_{ID|_{k-1}}, RL_{ID|_{k-1},T}, PP, \theta) \rightarrow \widetilde{TUK}$: It constructs the correlative component key $TUK_{ID|_{j-1},T,\theta} = (\{U_{i,0}, U_{i,1}, U_{i,2}, U_{i,3}\}_{i=0}^{k-1})$ to the node $\theta \in KUNode$ as follows: It chooses random exponents $y', r \in \mathbf{Z}_p$ and choose $\sigma_2, \psi_2 \in \mathbf{Z}_p$, then it constructs $\kappa_T^{sf}(T, y', r)$ of the first element-group $(U_{0,0}, U_{0,1}, U_{0,2}, U_{0,3})$ as

$$(w_0^{y'}(g_2g_3)^{y'\psi_2}, g^{y'}(g_2g_3)^{y'}, g^r, v_0^{y'}(g_2g_3)^{y'\sigma_2}(u_0^T h_0)^r)$$

And the contruction of the other element-groups follows the construction of RSK_{HIBE} and SK_{IBE,S_θ} in RHIBE.UpdateKey.

RHIBE.DeriveKeySF: Let $\widetilde{SK}_{ID_k}$ be a semi-functional secret key generated by the RHIBE.GenKeySF algorithm and $\widetilde{UK}_{ID|_{k-1},T}$ be a semi-functional update key for time T generated by the RHIBE.UpdateKeySF algorithm. If $ID|_k \notin RL_{ID|_{k-1}}$, then it finds a unique node θ^* by running CS.Match($CV_R(BT_{ID|_{k-1}}, RL_{ID|_{k-1}}, T)$, $Path(ID|_k)$). Otherwise, it outputs $\perp$. It derives $\widetilde{PSK}_{\theta^*} = (\{\widetilde{K}_{i,0}, \widetilde{K}_{i,1}, \widetilde{K}_{i,2}, \widetilde{K}_{i,3}\}_{i=1}^{k})$ from $\widetilde{SK}_{ID|_k}$ and $\widetilde{TUK}_{\theta^*} = (\{\widetilde{U}_{i,0}, \widetilde{U}_{i,1}, \widetilde{U}_{i,2}, \widetilde{U}_{i,3}\}_{i=0}^{k-1})$ from $\widetilde{UK}_{ID|_{k-1},T}$ for the node θ^*. Then the semi-functional decryption key $\widetilde{DK}_{ID|_k,T}$ is $\widetilde{DK}_{ID|_k,T} = (\{\widetilde{D}_{i,0}, \widetilde{D}_{i,1}, \widetilde{D}_{i,2}, \widetilde{D}_{i,3}\}_{i=0}^{k})$ as

$$\left(\widetilde{U}_{0,0}, \widetilde{U}_{0,1}, \widetilde{U}_{0,2}, \widetilde{U}_{0,3}, \{\widetilde{U}_{i,0}\widetilde{K}_{i,0}, \widetilde{U}_{i,1}\widetilde{K}_{i,1}, \widetilde{U}_{i,2}\widetilde{K}_{i,2}, \widetilde{U}_{i,3}\widetilde{K}_{i,3}\}_{i=1}^{k-1}, \widetilde{K}_{k,0}, \widetilde{K}_{k,1}, \widetilde{K}_{k,2}, \widetilde{K}_{k,3}\right)$$

Then we re-randomize it by running RHIBE.RandDK and output it.

4.2 Sequence of games

We define a squence of games to verify the advantage in distinguishing G_{Real} and G_{Final} is negligible. In Table 3, we give the types of key in the queries and the challenge cipertext in every game, and the decryption situation according to the types of keys and ciphertexts.

G_{Real}: It is the original game in which all seceret keys, update keys, decryption keys and ciphertexts are normal.

G_C: The challenge ciphertext is changed to be semi-functional and all other keys are still normal.

$G_{C'}$: This game is exactly like $Game_C$, except for a added restriction about the challenge key identity vector. We explain the restriction in Sec.4.6.

G_{E-S}: The secret keys are changed to ESF-2. The update keys and decryption keys are still normal. The challenge ciphertext is semi-functional. This game is used in the proof of the security against Type-1 adversary.

G_{E-U}: The update keys are changed to ESF-2. The secret keys and decryption keys are still normal. The challenge ciphertext is semi-functional. This game is used in the proof of the security against Type-2 adversary.

$G_{E-S'}$: This game is almost as same as G_{E-S} except the challenge ciphertext is chaged to ESF-1. This game is used in the proof of the security against Type-1 adversary.

$G_{E-U'}$: This game is almost as same as G_{E-U} where the update keys are ESF-2, the secret keys and decryption keys are normal, except the challenge ciphertext is chaged to ESF-1. This game is used in the proof of the security against Type-2 adversary.

$G_{ESF'}$: The update keys and secret keys are all changed to ESF-2. The challenge ciphertext is changed to ESF-1. The decryption keys are still normal.

$G_{SF''}$: All secret keys, update keys, and challenge ciphertext are changed to semi-functional. The decryption keys are still normal.

Table 3. Defination of games.

Games	Oracles	Key Types in Queries			Challenge Ciphertext			
		SK	UK	DK	*Normal*	*SF*	*ESF* – 1	*ESF* – 5
G_{real}		*Normal*	*Normal*	*Normal*	√			
G_C		*Normal*	*Normal*	*Normal*		√		
$G_{C'}$	O_0	*Normal*	*Normal*	*Normal*		√		
G_{E-S}	O_1	*ESF* – 2	*Normal*	*Normal*		√		
G_{E-U}	O_{1+}	*Normal*	*ESF* – 2	*Normal*				
$G_{E-S'}$	O_2	*ESF* – 2	*Normal*	*Normal*			◯	
$G_{E-U'}$	O_{2+}	*Normal*	*ESF* – 2	*Normal*				
$G_{ESF'}$	O_3	*ESF* – 2	*ESF* – 2	*Normal*			◯	
$G_{SF''}$	O_4	*SF*	*SF*	*Normal*		◯		
G_{E-D}	O_5	*SF*	*SF*	*ESF* – 2		◯		
G_{ESF}	O_6	*SF*	*SF*	*ESF* – 2			⊙	
$G_{SF'}$	O_7	*SF*	*SF*	*SF*		⊙		
G_{SF}		*SF*	*SF*	*SF*		⊙		

The decryption situation according to the type of keys and ciphertexts in different games is for the challenger $\mathcal{B}$ to check whether the keys are nominally semi-functional keys. √ means that the decryption key answered by the query $DKGen_Q$ and the derived decryption key from the corresponding secret key and update key outputed by $SKGen_Q$ and $KeyUp_Q$ both are able to decrypt the ciphertext. ◯ means that only the decryption key answered by the query $DKGen_Q$ is able to decrypt the ciphertext. ⊙ means that neither the queried decryption key nor the derived decryption key is able to decrypt the ciphertext.

https://doi.org/10.1371/journal.pone.0195204.t003

G_{E-D}: The decryption keys are changed to ESF-2. The other keys and the challenge ciphertext are still semi-functional.

G_{ESF}: The challenge ciphertext is changed to ESF-1. The update keys and secret keys are all still semi-functional. The decryption keys are still ESF-2.

$G_{SF'}$: The challenge ciphertext is changed to semi-functional. The decryption keys are changed to be semi-functional. That is, all secret keys, update keys, decryption keys, and challenge ciphertext are now semi-functional. This game is exactly like G_{SF}, except for a added restriction about the challenge key identity vector. We explain the restriction in Sec.4.6.

G_{SF}: The challenge ciphertext and all keys are semi-functional.

G_{Final}: The session key is changed to be random and so the adversary has no advantage to distinguish the challenge massage.

Let $Adv_{\mathcal{A}}^{RHIBE}$ be the advantage of $\mathcal{A}$ in the real game. From the all the lemmas in this section, we obtain the following equation

$$\begin{aligned} Adv_{\mathcal{A}}^{RHIBE}(\lambda) &\leq |Adv_{\mathcal{A}}^{G_{Real}}(\lambda) - Adv_{\mathcal{A}}^{G_C}(\lambda)| + |Adv_{\mathcal{A}}^{G_C}(\lambda) - Adv_{\mathcal{A}}^{G_{C'}}(\lambda)| \\ &\quad + |Adv_{\mathcal{A}}^{G_{C'}}(\lambda) - Adv_{\mathcal{A}}^{G_{SF'}}(\lambda)| + |Adv_{\mathcal{A}}^{G_{SF'}}(\lambda) - Adv_{\mathcal{A}}^{G_{SF}}(\lambda)| \\ &\quad + |Adv_{\mathcal{A}}^{G_{SF}}(\lambda) - Adv_{\mathcal{A}}^{G_{Final}}(\lambda)| \\ &\leq Adv_{\mathcal{B}}^{A1}(\lambda) + Adv_{\mathcal{B}}^{A2}(\lambda) \\ &\quad + (O(q_n log N_{max}) + O(q_n r_{max} log N_{max}) + O(l))(Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4}) \end{aligned}$$

4.3 Definition of oracles

We introduce seven oracles which answer queries from the challenger $\mathcal{B}$ by sampling various distributions of group elements from a composite order bilinear group. The outputs of Oracle

O_i will allow a simulator to produce different type of secret keys, update keys and decryption keys, different type of ciphertext and challenge keys for one corresponding game demonstrated in Table 3.

All oracles are defined with respect to a bilinear group G of order $p = p_1p_2p_3$ and initially choose random elements $g, u, v, w, u_0, v_0, w_0 \in G_{p_1}, g_2 \in G_{p_2}, g_3 \in G_{p_3}$ as well as random exponents $\psi_1, \psi_2, \sigma_1, \sigma_2, a', b', s, \delta_1, \delta_2, \gamma \in Z_n$. They provide the attacker with a description of the group G, as well as the group elements

$$\begin{gathered} g, u, v, w, g^s g_2^{\gamma}, w^y (g_2 g_3)^{y\psi_1}, g^y (g_2 g_3)^y, v^y (g_2 g_3)^{y\sigma_1}, \\ u_0, v_0, w_0, w_0^{y_0} (g_2 g_3)^{y_0\psi_2}, g^{y_0} (g_2 g_3)^{y_0}, v_0^{y_0} (g_2 g_3)^{y_0\sigma_2} \end{gathered} \tag{2}$$

Every oracle is allowed to simulate the semi-functional ciphertexts, normal and semi-functional (H)IBE private keys according to the provided group elements in Eq 2. We define the oracles from O_0 to O_4 in which the simulators will be allowed to produce a normal challenge decryption key. The outputs of Oracle O_0 will allow a simulator to produce a semi-functional challenge ciphertext, a normal challenge (H)IBE private key. The outputs of Oracle O_1 will allow a simulator to produce a semi-functional challenge ciphertext, a type-2 ephemeral semi-functional (ESF-2) challenge HIBE private key and a normal challenge IBE private key. The outputs of Oracle $O_{1'}$ will allow a simulator to produce a semi-functional challenge ciphertext, an type-2 ephemeral semi-functional (ESF-2) challenge IBE private key an normal challenge HIBE private key. The outputs of Oracle O_3 will allow a simulator to produce a type-1 ephemeral semi-functional(ESF-1) ciphertext, and a type-2 ephemeral semi-functional(ESF-2) challenge (H)IBE private key. Finally, the outputs of Oracle O_4 will allow a simulator to produce a semi-functional challenge ciphertext, and a semi-functional challenge (H)IBE private key.

We define the oracles from O_5 to O_7 in which the simulators will be allowed to produce a semi-functional challenge (H)IBE key. The outputs of Oracle O_5 will allow a simulator to produce a semi-functional ciphertext, and an ephemeral semi-functional challenge decryption key. The outputs of Oracle O_6 will allow a simulator to produce an type-1 ephemeral semi-functional(ESF-1) ciphertext, and a type-2 ephemeral semi-functional(ESF-2) challenge decryption key. Finally, the outputs of Oracle O_7 will allow a simulator to produce a semi-functional ciphertext, and a semi-functional challenge decryption key.

Oracle O_0 The first oracle, which we will denote by O_0, responds to queries as follows. Upon receiving a challenge HIBE-key-type query for $I \in Z_n$, it chooses $r, y' \in Z_n$ randomly and returns the group elements

$$(w^{y'}, g^{y'}, v^{y'} (u^I h)^r, g^r) \tag{3}$$

to the attacker. Upon receiving a challenge IBE-key-type query for $T \in Z_n$, it chooses $r', y'' \in Z_n$ randomly and returns the group elements

$$(w_0^{y''}, g^{y''}, v_0^{y''} (u_0^T h_0)^{r'}, g^{r'}) \tag{4}$$

to the attacker. Upon receiving a challenge decryption-key-type query for $I \in Z_n$ and $T \in Z_n$, it chooses $r, y', r', y'' \in Z_n$ randomly and returns the group elements

$$(w^{y'}, g^{y'}, v^{y'} (u^I h)^r, g^r, w_0^{y''}, g^{y''}, v_0^{y''} (u_0^T h_0)^{r'}, g^{r'}) \tag{5}$$

to the attacker. Upon receiving a ciphertext-type query for $I^* \in Z_n$, it chooses $t \in Z_n$ randomly and returns the group elements

$$(w^s g_2^{\delta_1} v^t, g^t, (u^{I^*} h)^t) \tag{6}$$

to the attacker. Upon receiving a ciphertext-type query for $T^* \in Z_n$, it chooses $t_0 \in Z_n$ randomly and returns the group elements

$$(w_0^s g_2^{\delta_2} v_0^{t_0}, g^{t_0}, (u_0^{T^*} h_0)^{t_0}) \tag{7}$$

to the attacker.

Oracle O_1 The next oracle, which we will denote by O_1, responds to queries as follows. Upon receiving a challenge HIBE-key-type query for $I \in Z_n$, it chooses $r'', y''' \in Z_n$ randomly, and also chooses $X_2, Y_2 \in G_{p_2}$, $X_3, Y_3 \in G_{p_3}$ randomly. It returns the group elements

$$(w^{y'''}, g^{y'''}, v^{y'''}(u^I h)^{r''} X_2 X_3, g^{r''} Y_2 Y_3) \tag{8}$$

to the attacker. It responds to a ciphertext-type query, a challenge IBE-key-type query and a challenge decryption-key-type query in the same way as O_0.

Oracle O_{1^+} The oracle O_{1^+} responds to queries as follows. Upon receiving a challenge IBE-key-type query for $T \in Z_n$, it chooses $r'', y''' \in Z_n$ randomly, and also chooses $X_2, Y_2 \in G_{p_2}$, $X_3, Y_3 \in G_{p_3}$ randomly. It returns the group elements

$$(w_0^{y'''}, g^{y'''}, v_0^{y'''}(u_0^T h_0)^{r''} X_2 X_3, g^{r''} Y_2 Y_3) \tag{9}$$

to the attacker. It responds to a ciphertext-type query, a challenge HIBE-key-type query and a challenge decryption-key-type query in the same way as O_0.

Oracle O_2 The next oracle, which we will denote by O_2, responds to queries as follows. Upon receiving a challenge HIBE-key-type query and a challenge IBE-key-type query, it responds in the same way as O_1. Upon receiving a ciphertext-type query for $I^* \in Z_n$, it chooses $t \in Z_n$ randomly and returns the group elements

$$(w^s g_2^{\delta_1} v^t g_2^{\sigma_1 t}, g^t g_2^t, (u^{I^*} h)^t g_2^{t(a'I^* + b')}) \tag{10}$$

to the attacker. Upon receiving a ciphertext-type query for $T^* \in Z_n$, it chooses $t_0 \in Z_n$ randomly and returns the group elements

$$(w_0^s g_2^{\delta_2} v_0^{t_0} g_2^{\sigma_2 t_0}, g^{t_0} g_2^{t_0}, (u_0^{T^*} h_0)^{t_0} g_2^{t_0(a'T^* + b')}) \tag{11}$$

to the attacker. It responds to a challenge decryption-key-type query in the same way as O_0.

Oracle O_{2^+} The next oracle, which we will denote by O_{2^+}, responds to queries as follows. Upon receiving a challenge HIBE-key-type query and a challenge IBE-key-type query, it responds in the same way as O_{1^+}. Upon receiving a ciphertext-type query for $I^* \in Z_n$, it chooses $t \in Z_n$ randomly and returns the group elements

$$(w^s g_2^{\delta_1} v^t g_2^{\sigma_1 t}, g^t g_2^t, (u^{I^*} h)^t g_2^{t(a'I^* + b')}) \tag{12}$$

to the attacker. Upon receiving a ciphertext-type query for $T^* \in Z_n$, it chooses $t_0 \in Z_n$ randomly and returns the group elements

$$(w_0^s g_2^{\delta_2} v_0^{t_0} g_2^{\sigma_2 t_0}, g^{t_0} g_2^{t_0}, (u_0^{T^*} h_0)^{t_0} g_2^{t_0(a'T^* + b')}) \tag{13}$$

to the attacker. It responds to a challenge decryption-key-type query in the same way as O_0.

Oracle O_3 The next oracle, which we will denote by O_3, responds to queries as follows. Upon receiving a challenge HIBE-key-type query and a ciphertext-type query, it responds in the same way as O_2. Upon receiving a challenge IBE-key-type query for $I \in Z_n$, it chooses r'', $y''' \in Z_n$ randomly, and also chooses $X_2, Y_2 \in G_{p_2}$, $X_3, Y_3 \in G_{p_3}$ randomly. It returns the group

elements

$$(w_0^{y'''}, g^{y'''}, v_0^{y'''}(u_0^T h_0)^{r''} X_2 X_3, g^{r''} Y_2 Y_3) \tag{14}$$

to the attacker. It responds to a challenge decryption-key-type query in the same way as O_0.

Oracle O_4 The next oracle, which we will denote by O_4, responds to ciphertext-type queries in the same way as O_0, and responds to a challenge HIBE-key-type query for $I \in Z_n$, by choosing $r, y' \in Z_n$ randomly and returns the group elements

$$(w^{y'}(g_2g_3)^{y'\psi_1}, g^{y'}(g_2g_3)^{y'}, v^{y'}(g_2g_3)^{y'\sigma_1}(u^I h)^r, g^r) \tag{15}$$

to the attacker. Upon receiving a challenge IBE-key-type query for $T \in Z_n$, it chooses $r', y'' \in Z_n$ randomly and returns the group elements

$$(w_0^{y''}(g_2g_3)^{y''\psi_2}, g^{y''}(g_2g_3)^{y''}, v_0^{y''}(g_2g_3)^{y''\sigma_2}(u_0^T h_0)^{r'}, g^{r'}) \tag{16}$$

to the attacker. It responds to a challenge decryption-key-type query in the same way as O_0.

Oracle O_5 The next oracle, which we will denote by O_5, responds to queries as follows. Upon receiving a challenge decryption-key-type query for $I, T \in Z_n$, it chooses $r, y', r', y'' \in Z_n$ randomly, and also chooses $X_2, Y_2, X_2', Y_2' \in G_{p_2}, X_3, Y_3, X_3', Y_3' \in G_{p_3}$ randomly. It returns the group elements

$$(w^{y'}, g^{y'}, v^{y'}(u^I h)^r X_2 X_3, g^r Y_2 Y_3, w_0^{y''}, g^{y''}, v_0^{y''}(u_0^T h_0)^{r'} X_2' X_3', g^{r'} Y_2' Y_3') \tag{17}$$

to the attacker. It responds to a ciphertext-type query and a challenge (H)IBE-key-type query in the same way as O_4.

Oracle O_6 The next oracle, which we will denote by O_6, responds to queries as follows. Upon receiving a ciphertext-type query for $I^* \in Z_n$, it chooses $t \in Z_n$ randomly and returns the group elements

$$(w^s g_2^{\delta_1} v^t g_2^{\sigma_1 t}, g^t g_2^t, (u^{I^*} h)^t g_2^{t(a'I^* + b')}) \tag{18}$$

to the attacker. Upon receiving a ciphertext-type query for $T^* \in Z_n$, it chooses $t_0 \in Z_n$ randomly and returns the group elements

$$(w_0^s g_2^{\delta_2} v_0^{t_0} g_2^{\sigma_2 t_0}, g^{t_0} g_2^{t_0}, (u_0^{T^*} h_0)^{t_0} g_2^{t_0(a'T^* + b')}) \tag{19}$$

to the attacker. It responds to a decryption-type query and a challenge (H)IBE-key-type query in the same way as O_5.

Oracle O_7 The last oracle, which we will denote by O_7, responds to ciphertext-type queries in the same way as O_0, and responds to a challenge decryption-key-type query for $I, T \in Z_n$, by choosing $r, y', r', y'' \in Z_n$ randomly and returns the group elements

$$(w^{y'}(g_2g_3)^{y'\psi_1}, g^{y'}(g_2g_3)^{y'}, v^{y'}(g_2g_3)^{y'\sigma_1}(u^I h)^r, g^r,$$
$$w_0^{y''}(g_2g_3)^{y''\psi_2}, g^{y''}(g_2g_3)^{y''}, v_0^{y''}(g_2g_3)^{y''\sigma_2}(u_0^T h_0)^{r'}, g^{r'})$$

to the attacker. It responds to a challenge (H)IBE-key-type query in the same way as O_6.

We define the advantage of an attacker $\mathcal{A}$ in distinguishing between O_i and O_j to be $|Pr[\mathcal{A}(O_i) = 1] - Pr[\mathcal{A}(O_j) = 1]|$. Here, we assume that $\mathcal{A}$ interacts with either O_i or O_j, and then outputs a bit 0 or 1 encoding its guess of which oracle it interacted with.

4.4 Indistinguishability of $G_{C'}$ and $G_{SF''}$

4.4.1 Strategy for the indistinguishability of $G_{C'}$ and $G_{SF''}$. For the proof of the indistinguishability of $G_{C'}$ and $G_{SF'}$, we cannot use the simple nested dual system in U-HIBE [11] that change a normal private key(or normal update key) to an ephemeral semi-fuctional private key(or semi-functional update key) one by one since the adversary of RHIBE can query a private key for $ID|_k \in Prefix(ID^*|_l)$ and an update key for T^*.

To solve this problem, we firstly use a modular design strategy like [13] and construct the private keys and update keys from smaller component keys. A secret key $SK_{ID|_k}$ consists of many HIBE private keys which are represented as $\{SK_{HIBE,S_\theta}\}_{S_\theta \in Path}$ and an update key $UK_{ID|_{k-1},T,R}$ consists a randomized decryption key RSK_{HIBE} and many IBE private keys $\{SK_{IBE,S_i}\}_{S_i \in CVR}$ where each HIBE private key (or an IBE private key) is associated with a node S_j in $BT_{ID|_{k-1}}$. The HIBE and IBE private keys can be grouped together if they are related to the same node S_j in $BT_{ID|_{k-1}}$ and a correct decryption key is constructed form the grouped (H)IBE private key.

To uniquely identify a node $S_j \in BT_{ID|_{k-1}}$, we define a node identifier NID of this node as a string $ID|_{k-1}||L_j$ where $L_j = Label(v_j)$. To prove the indistinguishability of $G_{C'}$ and $G_{SF''}$, we change normal HIBE private keys and normal IBE private keys that are related to the same node identifier NID into (ephemeral) semi-functional keys by defining additional hybrid games. This additional hybrid games are performed for all node identifiers that are used in the key queries of the adversary.

Secondly, we give the equivalent model in which the challenger $\mathcal{B}$ answers the secret (update, and decryption) key queries of the adversery $\mathcal{A}$ by requesting the associated (H)IBE private keys from an oracle simulator $\mathcal{O}$, shown in Fig 1. When the adversary $\mathcal{A}$ queries $\mathcal{B}$ for the secret key, update key or decryption key for some identity and some time period, B constructs the key by the (H)IBE-challenge-key or decryption-challenge-key it queries from the oracle simulator $\mathcal{O}$. $\mathcal{O}$ adaptively answers $\mathcal{B}$ the corresponding group elements which it constructs by using the public paremeters given by some complexity assumption. Therefore, under the complexity assumptions, the oracle O_i that $\mathcal{O}$ chooses to answer $\mathcal{B}$ is indistinguishable and consequently the adversary $\mathcal{A}$ cannot distinguish whether $\mathcal{A}$ is playing the real RHIBE game or other variation games based on all the answers $\mathcal{A}$ recieves after the adaptive queries to $\mathcal{B}$.

For additional hybrid games that change HIBE private keys (or IBE private keys) that are related to the same node identifier $NID = ID|_{k-1}||L_j$ from normal keys to semi-functional keys,

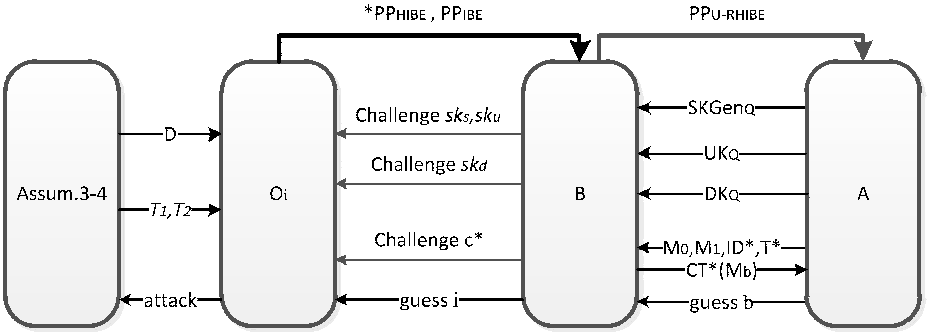

Fig 1. The query process in the proof of the indistinguishability of $G_{C'}$ and $G_{SF'}$. *The group elements that the oracle simulator gives to the challenger $\mathcal{B}$ are not only the public parameters PP_{HIBE} and PP_{IBE}, but also the group elements for constructing the (ephemeral) semi-functional keys and ciphertexts and the public elements given by the assumptions.

https://doi.org/10.1371/journal.pone.0195204.g001

we need to define an index pair (i_n, i_c) for an HIBE private key (or an IBE private key) that is related to the node $v_j \in BT_{ID|_{k-1}}$ where i_n is a node index and i_c is a counter index. Suppose that an HIBE private key (or an IBE private key) is related to a node *NID*. The node index i_n for the HIBE private key (or the IBE private key) is assigned as follows: If the node $v_j \in BT_{ID|_{k-1}}$ with a node identifier *NID* appears first time in key queries, then we set in as the number of distinct node identifiers in previous key queries plus one. If the node identifier *NID* already appeared before in key queries, then we set i_n as the value i'_n of previous HIBE private key (or IBE private key) with the same node identifier. The counter index i_c of an HIBE private key is assigned as follows: If the node identifier NID appears first time in HIBE private key queries, then we set i_c as one. If the node identifier NID appeared before in HIBE private key queries, then we set i_c as the number of HIBE private keys with the same node identifier that appeared before plus one. Similarly, we assigns the counter index i_c of an IBE private key.

Thirdly, we divide the behavior of an adversary as two types: Type-1 and Type-2. We next show that the semi-functional key invariance property holds for two types of the adversary. Let ID^*_l be the challenge hierarchical identity and T^* be the challenge time. For a challenge node v with the node index h in the hybrid games from $Game_C$ and $Game_{SF}$, the adversary types are formally defined as follows:

1. **Type-1**: An adversary is Type-1 if it queries on a hierarchical identity $ID|_k \notin Prefix(ID^*_l)$ for all HIBE private keys with the node index h, and it queries on time $T = T^*$ for at least one IBE private key with the node index h.
2. **Type-2**: An adversary is Type-2 if it queries on time $T \notin T^*$ for all IBE private keys with the node index h. Note that it may query on a hierarchical identity $ID|_k \in Prefix(ID^*|_l)$ for at least one HIBE private key with h, or it may query on a hierarchical identity $ID|_j \notin Prefix(ID^*_l)$ for all HIBE private keys with h.

We prove our dual system encryption RHIBE scheme via a hybrid argument over the sequence of games in Table 3. For the different type of adversary, the squence of games is basicly the same except that:

1. For the Type-1 adversary, we prove the indistinguishability of $G_{C'}$ and $G_{ESF'}$ by the transition from $G_{C'}$ to G_{EK-S}, and to $G_{ESF'}$ without the attacker's advantage changing by a non-negligible amount.
2. For the Type-2 adversary, we prove the indistinguishability of $G_{C'}$ and $G_{ESF'}$ by the transition from $G_{C'}$ to G_{EK-U}, and to $G_{ESF'}$ without the attacker's advantage changing by a non-negligible amount.

Theorem 2 *Under Assumptions 3 and 4, our dual system encryption RHIBE scheme has the equation*

$$\begin{aligned} |Adv_{\mathcal{A}}^{G_{C'}}(\lambda) - Adv_{\mathcal{A}}^{G_{SF''}}(\lambda)| \leq (O(q_n log N_{max}) + O(q_n r_{max} log N_{max}))(Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4}) \\ + O(l)(Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4}) \end{aligned} \tag{20}$$

We will prove these indistinguishabilities between games $G_{C'}$, G_{E-S} (or G_{E-U}), $G_{E-S'}$ (or $G_{E-U'}$), $G_{ESF'}$, and $G_{SF''}$ by going through several intermediary oracles. The main properties of our oracles are summarized in Tables 4 and 5 for the Type-1 adversary and Table 6 for the Type-2 adversary respectively. We intend these tables to be used only as a quick reference guide, not as a definition. We give a complete proof for the Type-1 adversary, and a brief explanation of the proof for the Type-2 adversary is demonstrated then.

Table 4. Simulation of challenge keys and cipertext in oracles for the proof of the indistinguishability between $G_{C'}$ and $G_{SF''}$ under Type-1 adversary.

Oracle	CT-Type Response	SK-Type Response	UK-Type Response	DK-Type Response
$\boldsymbol{O_0}$	SF	Normal	Normal	Normal
$O_{1/2}$	SF	ESF-1	Normal	Normal
$\boldsymbol{O_1}$	SF	ESF-2	Normal	Normal
O_i^*	ESF-2^i	ESF-2	Normal	Normal
O_i'	ESF-3^i	ESF-2	Normal	Normal
O_i''	ESF-4^i	ESF-2	Normal	Normal
$\boldsymbol{O_2}$	ESF-1	ESF-2	Normal	Normal
$O_{5/2}$	ESF-1	ESF-2	ESF-1	Normal
$\boldsymbol{O_3}$	ESF-1	ESF-2	ESF-2	Normal
$O_{3.1}$	ESF-1	ESF-3	ESF-2	Normal
$O_{3.2}$	ESF-1	ESF-3	ESF-3	Normal
†$O_{3.3}$	ESF-5	ESF-3	ESF-3	Normal
†$O_{3.4}$	ESF-5	ESF-4	ESF-3	Normal
†$O_{3.5}$	ESF-5	ESF-4	ESF-4	Normal
$O_{3.6}$	ESF-1	ESF-4	ESF-4	Normal
$O_{7/2'}$	ESF-1	ESF-4	ESF-5	Normal
$\widetilde{O}_i^*$	ESF-2^i	ESF-4	SF	Normal
$\widetilde{O}_i'$	ESF-3^i	ESF-4	SF	Normal
$\widetilde{O}_i''$	ESF-4^i	ESF-4	SF	Normal
$O_{7/2}$	SF	ESF-5	SF	Normal
$\boldsymbol{O_4}$	SF	SF	SF	Normal

Note: oracles marked with † initialize with an extra G_{p_3} term on $g^s g_2^\gamma$.

https://doi.org/10.1371/journal.pone.0195204.t004

4.4.2 Type-1 adversary. As defined before, the Type-1 adversary is restricted to queries on a hierarchical identity $ID|_k \notin Prefix(ID_l^*)$. By quering for all HIBE private keys with any node index h where the node is on the path from the root to the leaf node $v_{ID|_k}$ in the tree $BT_{ID|_{k-1}}$, the adversary derives the secret key of $ID|_k$.

So we could show an information theoretic argument for the HIBE private keys from normal to ephemeral semi-functional HIBE keys, then to semi-functional HIBE keys. At the

Table 5. Defination of games between G_{ESF} and $G_{SF''}$.

Games	Oracles	Keys in Queries			Challenge Ciphertext			
		SK	UK	DK	*Normal*	*SF*	*ESF* − 1	*ESF* − 5
G_{ESF}	O_3	*ESF* − 2	*ESF* − 2	*Normal*			○	
G_{ESF-1}	$O_{3.1}$	*ESF* − 3	*ESF* − 2	*Normal*			○	
G_{ESF-2}	$O_{3.2}$	*ESF* − 3	*ESF* − 3	*Normal*			○	
G_{ESF-3}	$O_{3.3}$	*ESF* − 3	*ESF* − 3	*Normal*				○
G_{ESF-4}	$O_{3.4}$	*ESF* − 4	*ESF* − 3	*Normal*				○
G_{ESF-5}	$O_{3.5}$	*ESF* − 4	*ESF* − 4	*Normal*				○
G_{ESF-6}	$O_{3.6}$	*ESF* − 4	*ESF* − 4	*Normal*			○	
G_{ESF-7}	$\widetilde{O}_{q_c}^*$	*ESF* − 4	*SF*	*Normal*			○	
G_{ESF-8}	$\widetilde{O}_0^*$	*ESF* − 4	*SF*	*Normal*		○		
$G_{SF''}$	O_4	*SF*	*SF*	*Normal*		○		

https://doi.org/10.1371/journal.pone.0195204.t005

Table 6. Simulation of challenge keys and cipertext in oracles under Type-2 adversary for the proof of the indistinguishability between $G_{C'}$ and $G_{SF''}$.

Oracle	CT-Type Response	SK-Type Response	UK-Type Response	DK-Type Response
$\boldsymbol{O_0}$	SF	Normal	Normal	Normal
$O_{1/2^+}$	SF	Normal	ESF-1	Normal
$\boldsymbol{O_{1^+}}$	SF	Normal	ESF-2	Normal
$O^*_{i^+}$	ESF-2^i	Normal	ESF-2	Normal
O'_{i^+}	ESF-3^i	Normal	ESF-2	Normal
O''_{i^+}	ESF-4^i	Normal	ESF-2	Normal
$\boldsymbol{O_{2^+}}$	ESF-1	Normal	ESF-2	Normal
$O_{5/2^+}$	ESF-1	ESF-1	ESF-2	Normal
$\boldsymbol{O_3}$	ESF-1	ESF-2	ESF-2	Normal
$O_{3.1^+}$	ESF-1	ESF-2	ESF-3	Normal
$O_{3.2}$	ESF-1	ESF-3	ESF-3	Normal
$\dagger O_{3.3}$	ESF-5	ESF-3	ESF-3	Normal
$\dagger O_{3.4^+}$	ESF-5	ESF-3	ESF-4	Normal
$\dagger O_{3.5}$	ESF-5	ESF-4	ESF-4	Normal
$O_{3.6}$	ESF-1	ESF-4	ESF-4	Normal
$O_{3.7^+}$	ESF-1	ESF-5	ESF-4	Normal
$\widetilde{O}^*_{i^+}$	ESF-2^i	SF	ESF-4	Normal
$\widetilde{O}'_{i^+}$	ESF-3^i	SF	ESF-4	Normal
$\widetilde{O}''_{i^+}$	ESF-4^i	SF	ESF-4	Normal
$O_{7/2^+}$	SF	SF	ESF-5	Normal
$\boldsymbol{O_4}$	SF	SF	SF	Normal

Note: oracles marked with † initialize with an extra G_{p_3} term on $g^s g_2^\gamma$.

https://doi.org/10.1371/journal.pone.0195204.t006

meanwhile, by adaptively transforming the types of IBE private keys sooner or later than the transformation of HIBE private keys, we avoid a potential paradox for the update keys.

From the flollowing Lemma 1, to Lemma 20, we obtain the advantage of Type-1 adversary to distinguish between $G_{C'}$ and $G_{SF'}$ under Type-1 adversary as

$$\begin{aligned} Adv_{\mathcal{A}}^{G_{C'}} - Adv_{\mathcal{A}}^{G_{SF''}} &\leq |Adv_{\mathcal{A}}^{G_{C'}} - Adv_{\mathcal{A}}^{G_{E-S}}| + |Adv_{\mathcal{A}}^{G_{E-S}} - Adv_{\mathcal{A}}^{G_{E-S'}}| \\ &+ |Adv_{\mathcal{A}}^{G_{E-S'}} - Adv_{\mathcal{A}}^{G_{ESF'}}| + |Adv_{\mathcal{A}}^{G_{ESF'}} - Adv_{\mathcal{A}}^{G_{SF''}}| \\ &\leq (O(q_n(q_s + q_e)) + O(l))(Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4}) \\ &\leq (O(q_n log N_{max} + q_n r_{max} log N_{max}) + O(l))(Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4}) \end{aligned} \tag{21}$$

We give the proof of those lemmas in Appendix.B.

(1) Indistinguishability of $G_{C'}$ and G_{E-S}

For the security proof of the indistinguishability of $G_{C'}$ and G_{E-S}, we define a sequence of additional hybrid games $G_{C',1}, \ldots, G_{C',h}, \ldots, G_{C',q_n}$, where $G_{C'} = G_{C',0}$ and q_n is the number of all node identifiers that are used in HIBE private keys and IBE private keys of an adversary. In the game $G_{C',h}$ for $1 \leq h \leq q_n$, the challenge ciphertext is semi-functional, all IBE private keys are normal, HIBE private keys with a node index $i_n \leq h$ are of ESF-2, the remaining HIBE private keys with a node index $i_n > h$ are normal.

Oracle $O_{1/2}$ This oracle initializes in the same way as O_0, O_1 and provides the attacker with initial group elements from the same distribution. Upon receiving a challenge HIBE-key-type

query for $I \in Z_n$, it chooses $r', y' \in Z_n$ randomly, and also chooses X_3, $Y_3 \in G_{p_3}$ randomly. It returns the group elements

$$(w^{y'}, g^{y'}, v^{y'}(u^I h)^{r'} X_3, g^{r'} Y_3) \tag{22}$$

to the attacker. It responds to a ciphertext-type query or a challenge IBE-key-type query in the same way as O_0.

We define hybrid games $H_{1,1}$, $H_{1,2}$, $\cdots$, $H_{h_c,1}$, $H_{h_c,2}$, $\ldots$, $H_{q_s,1}$, $H_{q_s,2}$ where $H_{0,2} = G_{C',h}$ and $H_{q_s,2} = G_{C',h+1}$, and q_s is the maximun number of HIBE private key queries for the node index h. The games are formally defined as follows:

Game $H_{h_c,1}$ This game $H_{h_c,1}$ for $1 \leq h_c \leq q_s$ is almost the same as $G_{2,h-1}$ except the generation of HIBE private keys and IBE private keys with the node θ_h of the index h. An IBE private key with an index pair (h, i_c) is generated as normal. An HIBE private key with an index pair (h, i_c) is generated as follows:

1. $i_c < h_c$: It generates a ESF-2 SK_{HIBE,θ_h}.
2. $i_c = h_c$: It generates a ESF-1 SK_{HIBE,θ_h} by using the element groups in Eq 22.
3. $i_c > h_c$: It simply generates a normal HIBE private key.

Game $H_{h_c,2}$ This game $H_{h_c,1}$ for $1 \leq h_c \leq q_s$ is almost the same as $H_{h_c,1}$ except the generation of HIBE private key with an index pair (h, i_c) and $i_c = h_c$ is generated as a ESF-2 SK_{HIBE,θ_h} by using the element groups in Eq 8.

Lemma 1 *Under Assumptions 3, no PPT attacker can distinguish between* O_0 *and* $O_{1/2}$ *with non-negligible advantage. So no PPT attacker can distinguish between* $H_{h_c-1,2}$ *and* $H_{h_c,1}$ *with non-negligible advantage.*

Lemma 2 *Under Assumptions 4, no PPT attacker can distinguish between* $O_{1/2}$ *and* O_1 *with non-negligible advantage. So no PPT attacker can distinguish between* $H_{h_c,1}$ *and* $H_{h_c,2}$ *with non-negligible advantage.*

Let $Adv_{\mathcal{A}}^{G_{C',h}}$ be the advantage of $\mathcal{A}$ in a game $G_{C',h}$. From the Lemma 1, 2, we obtain the following equation

$$\begin{aligned} Adv_{\mathcal{A}}^{G_{C',h-1}} - Adv_{\mathcal{A}}^{G_{C',h}} &\leq \textstyle\sum_{h_c=1}^{q_s} (|Adv_{\mathcal{A}}^{H_{h_c-1,2}} - Adv_{\mathcal{A}}^{H_{h_c,1}}| + |Adv_{\mathcal{A}}^{H_{h_c,1}} - Adv_{\mathcal{A}}^{H_{h_c,2}}|) \\ &\leq O(q_s)(Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4}) \end{aligned}$$

So we obtain the following equation

$$Adv_{\mathcal{A}}^{G_{C'}} - Adv_{\mathcal{A}}^{G_{E-S}} \leq \sum_{h=1}^{q_n} (Adv_{\mathcal{A}}^{G_{C',h-1}} - Adv_{\mathcal{A}}^{G_{C',h}}) \leq O(q_n q_s)(Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4}) \tag{23}$$

(2) Indistinguishability of G_{E-S} and $G_{E-S'}$

We now prove the indistinguishability of G_{E-S} and $G_{E-S'}$ in a hybrid argument using polynomially many steps. We let q_c denote the number of ciphertext-type queries made by a PPT attacker $\mathcal{A}$. Firstly we define hybrid games $S_{-1,1}$, $S_{0,2}$, $S_{0,3}$, $S_{0,1}$, $S_{1,2}$, $S_{1,3}$, $S_{1,1}\cdots$, $S_{k,2}$, $S_{k,3}$, $S_{k,1}$, $\ldots$, $S_{q_c-1,2}$, $S_{q_c-1,3}$, $S_{q_c-1,1}$, where $S_{-1,1} = G_{E-S}$ and $S_{q_c-1,1} = G_{E-S'}$. The games are formally defined as follows:

Game $S_{k,1}$ This game $S_{k,1}$ for $0 \leq k \leq q_c$ is almost the same as G_{E-S} except the generation of the challenge ciphertext. The challenge ciphertext of $(T^*, I_1^*, \ldots, I_{q_c-1}^*)$ is generated as EST-2^k-CT outputed by **EncryptESF-2^k** defined in AppendixA.

Game $S_{k,2}$ This game $S_{k,2}$ for $0 \leq k \leq q_c - 1$ is almost the same as G_{E-S} except the generation of the challenge ciphertext. The challenge ciphertext of $(T^*, I_1^*, \ldots, I_{q_c-1}^*)$ is generated as EST-3^k-CT outputed by **EncryptESF-3**^{k} defined in AppendixA.

Game $S_{k,3}$ This game $S_{k,3}$ for $0 \leq k \leq q_c - 1$ is almost the same as G_{E-S} except the generation of the challenge ciphertext. The challenge ciphertext of $(T^*, I_1^*, \ldots, I_{q_c-1}^*)$ is generated as EST-4^k-CT outputed by **EncryptESF-4**^{k} defined in AppendixA.

We will define additional oracles O_i^* for each i from 0 to $q_c - 1$, O_i' for each i from 0 to $q_c - 1$, and O_i'' for each i from 0 to $q_c - 1$, to sample various distributions of group elements used for constructing the various types of ciphertexts in Game $S_{k,1}$, Game $S_{k,2}$ and Game $S_{k,3}$.

Oracle O_i^* This oracle initializes in the same way as O_1, O_2 and provides the attacker with initial group elements from the same distribution. It also responds to challenge key-type queries in the same way as O_1, O_2. It keeps a counter of ciphertext-type queries which is initially equal to zero. It increments this counter after each response to a ciphertext-type query. In response to the jth ciphertext-type query for some I_j^*, if $j \leq i$, it responds exactly like O_2. If $j > i$, it responds exactly like O_1. In particular, O_0^* is identical to O_1 and O_q^* is identical to O_2.

Oracle O_i' This oracle acts the same as O_i^* except in its response to the i^{th} ciphertext-type query. For the i^{th} ciphertext-type query for identity I^*, it chooses a random $t \in Z_N$ and random elements X_3, $Y_3 \in G_{p_3}$ and responds with:

$$(w^s g_2^{\delta_1} v^t X_3^{\sigma_1}, g^t X_3, (u^{I_j^*} h)^t Y_3) \tag{24}$$

If $i = 0$, the i^{th} ciphertext-type query is for time T^*. It chooses a random $t_0 \in Z_N$ and random elements X_3', $Y_3' \in G_{p_3}$ and responds with:

$$(w_0^s g_2^{\delta_2} v_0^{t_0} X_3'^{\sigma_2}, g^{t_0} X_3', (u_0^{T^*} h_0)^{t_0} Y_3') \tag{25}$$

Oracle O_i'' This oracle acts the same as O_i^* except in its response to the i^{th} ciphertext-type query. For the i^{th} ciphertext-type query for identity I^*, it chooses a random $t \in Z_N$ and random elements X_3, $Y_3 \in G_{p_3}$ and responds with:

$$(w^s g_2^{\delta_1} v^t X_3^{\sigma_1}, g^t g_2^t X_3, (u^{I_j^*} h)^t g_2^{t(a' I_j^* + b')} Y_3) \tag{26}$$

If $i = 0$, the i^{th} ciphertext-type query is for time T^*. It chooses a random $t_0 \in Z_N$ and random elements X_3', $Y_3' \in G_{p_3}$ and responds with:

$$(w_0^s g_2^{\delta_2} v_0^{t_0} X_3'^{\sigma_2}, g^{t_0} g_2^{t_0} X_3', (u_0^{T^*} h_0)^{t_0} g_2^{t_0(a' I_j^* + b')} Y_3') \tag{27}$$

Lemma 3 *Under Assumptions 3, no PPT attacker can distinguish between* O_{k-1}^* *and* O_k' *with non-negligible advantage. So no PPT attacker can distinguish between* $S_{k-1,1}$ *and* $S_{k,2}$ *with non-negligible advantage.*

Lemma 4 *Under Assumptions 4, no PPT attacker can distinguish between* O_k' *and* O_k'' *with non-negligible advantage. So no PPT attacker can distinguish between* $S_{k,2}$ *and* $S_{k,3}$ *with non-negligible advantage.*

Lemma 5 *Under Assumptions 3, no PPT attacker can distinguish between* O_k'' *and* O_k^* *with non-negligible advantage. So no PPT attacker can distinguish between* $S_{k,3}$ *and* $S_{k,1}$ *with non-negligible advantage.*

Let $Adv_{\mathcal{A}}^{S_{k,1}}$, $Adv_{\mathcal{A}}^{S_{k,2}}$ and $Adv_{\mathcal{A}}^{S_{k,3}}$ be the advantage of $\mathcal{A}$ in the games $S_{k,1}$, $S_{k,2}$ and $S_{k,3}$. From the Lemma 3, 4, 5, we obtain the following equation

$$\begin{aligned} Adv_{\mathcal{A}}^{G_{E-S}} - Adv_{\mathcal{A}}^{G_{E-S'}} &\leq \sum_{k=0}^{q_c-1}(|Adv_{\mathcal{A}}^{S_{k-1,1}} - Adv_{\mathcal{A}}^{S_{k,2}}| + |Adv_{\mathcal{A}}^{S_{k,2}} - Adv_{\mathcal{A}}^{S_{k,3}}| \\ &\quad + |Adv_{\mathcal{A}}^{S_{k,3}} - Adv_{\mathcal{A}}^{S_{k,1}}|) \\ &\leq q_c(2Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4}) \end{aligned} \tag{28}$$

(3) Indistinguishability of $G_{E-S'}$ and $G_{ESF'}$

For the security proof of the indistinguishability of $G_{E-S'}$ and $G_{ESF'}$, we define a sequence of additional hybrid games $G_{S',1}, \ldots, G_{S',h}, \ldots, G_{S',q_n}$, where $G_{E-S'} = G_{S',0}$ and q_n is the number of all node identifiers that are used in HIBE private keys and IBE private keys of an adversary. In the game $G_{S',h}$ for $1 \leq h \leq q_n$, the challenge ciphertext is semi-functional, all IBE private keys are ESF-2, IBE private keys with a node index $i_n \leq h$ are of ESF-2, the remaining HIBE private keys with a node index $i_n > h$ are normal.

Oracle $O_{5/2}$ This oracle initializes in the same way as O_2, O_3 and provides the attacker with initial group elements from the same distribution. Upon receiving a challenge IBE-key-type query for $T \in Z_n$, it chooses $r', y' \in Z_n$ randomly, and also chooses $X_3, Y_3 \in G_{p_3}$ randomly. It returns the group elements

$$(w_0^{y'}, g^{y'}, v_0^{y'}(u_0^T h_0)^{r'} X_3, g^{r'} Y_3) \tag{29}$$

to the attacker. It responds to a ciphertext-type query or a challenge HIBE-key-type query in the same way as O_2.

We define hybrid games $E_{1,1}, E_{1,2}, \cdots, E_{h_c,1}, H_{h_c,2}, \ldots, E_{q_s,1}, E_{q_s,2}$ where $E_{0,2} = G_{S',h}$ and $E_{q_e,2} = G_{S',h+1}$, and q_e is the maximun number of IBE private key queries for the node index h. The games are formally defined as follows:

Game $E_{h_c,1}$ This game $E_{h_c,1}$ for $1 \leq h_c \leq q_e$ is almost the same as $G_{S',h}$ except the generation of HIBE private keys and IBE private keys with the node index h. An HIBE private key with an index pair (h, i_c) is generated as ESF-2. An IBE private key with an index pair (h, i_c) is generated as follows:

1. $i_c < h_c$: It generates a normal $SK'_{IBE,h}$ and converts the key to a ESF-2 $SK_{IBE,h}$.
2. $i_c = h_c$: It generates a normal $SK'_{IBE,h}$ and converts the key to a ESF-1 $SK_{IBE,h}$ by using the element groups in Eq 29.
3. $i_c > h_c$: It simply generates a normal IBE private key.

Game $E_{h_c,2}$ This game $E_{h_c,2}$ for $1 \leq h_c \leq q_e$ is almost the same as $E_{h_c,1}$ except the generation of IBE private key with an index pair (h, i_c) and $i_c = h_c$ is generated as a ESF-2 $SK_{IBE,h}$ by using the element groups in Eq 14.

Lemma 6 *Under Assumptions 3, no PPT attacker can distinguish between O_2 and $O_{5/2}$ with non-negligible advantage. So no PPT attacker can distinguish between $E_{h_c-1,2}$ and $E_{h_c,1}$ with non-negligible advantage.*

Lemma 7 *Under Assumptions 4, no PPT attacker can distinguish between $O_{5/2}$ and O_3 with non-negligible advantage. So no PPT attacker can distinguish between $E_{h_c,1}$ and $E_{h_c,2}$ with non-negligible advantage.*

Let $Adv_{\mathcal{A}}^{G_{E',h}}$ be the advantage of $\mathcal{A}$ in a game $G_{E',h}$. From the Lemma 6, 7, we obtain the following equation

$$Adv_{\mathcal{A}}^{G_{E',h-1}} - Adv_{\mathcal{A}}^{G_{E',h}} \leq \sum_{h_c=1}^{q_e}(|Adv_{\mathcal{A}}^{E_{h_c,1}} - Adv_{\mathcal{A}}^{E_{h_c,2}}| + |Adv_{\mathcal{A}}^{E_{h_c-1,2}} - Adv_{\mathcal{A}}^{E_{h_c,1}}|)$$
$$\leq O(q_e)(Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4})$$

So we obtain the following equation

$$Adv_{\mathcal{A}}^{G_{E-S'}} - Adv_{\mathcal{A}}^{G_{ESF'}} \leq \sum_{h=1}^{q_n}(Adv_{\mathcal{A}}^{G_{E',h-1}} - Adv_{\mathcal{A}}^{G_{E',h}}) \leq O(q_n q_e)(Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4}) \quad (30)$$

(4) Indistinguishability of $G_{ESF'}$ and $G_{SF''}$

For the security proof of the indistinguishability of $G_{ESF'}$ and $G_{SF''}$, we define a sequence of games $G_{ESF'-1}, \cdots, G_{ESF'-5}$ to change the type of secret keys and update keys from ESF-2 to ESF-4 and the type of ciphertexts from ESF-1 to ESF-5 and $G_{ESF'-6}, \cdots, G_{ESF'-8}$ to change the type of update keys to semi-functional and the type of ciphertexts back to semi-functional. In Table 5, we give the types of key in the queries and the challenge cipertext in every game, and the decryption situation according to the types of keys and ciphertexts.

$\mathbf{G_{ESF'-1}}$: The secret keys are changed to ESF-3. The update keys are still ESF-2. The challenge ciphertext is still ESF-1. The decryption keys are still normal.

$\mathbf{G_{ESF'-2}}$: The update keys are changed to ESF-3. The secret keys are ESF-3. The challenge ciphertext is still ESF-1. The decryption keys are still normal.

$\mathbf{G_{ESF'-3}}$: The challenge ciphertext is changed to ESF-5. The secret keys and the update keys are still ESF-3. The decryption keys are still normal.

$\mathbf{G_{ESF'-4}}$: The secret keys are changed to ESF-4. The update keys are still ESF-3. The challenge ciphertext is ESF-5. The decryption keys are still normal.

$\mathbf{G_{ESF'-5}}$: The update keys are changed to ESF-4. The secret keys are ESF-4. The challenge ciphertext is ESF-5. The decryption keys are still normal.

$\mathbf{G_{ESF'-6}}$: The challenge ciphertext is changed to ESF-1. The secret keys and the update keys are ESF-4. The decryption keys are still normal.

$\mathbf{G_{ESF'-7}}$: The update keys are changed to semi-functional update keys. The secret keys are ESF-4. The challenge ciphertext is ESF-1. The decryption keys are still normal.

$\mathbf{G_{ESF'-8}}$: The challenge ciphertext is changed to semi-functional. The secret keys are ESF-4. The update keys are semi-functional update keys. The decryption keys are still normal.

We firstly prove the indistinguishabilities between $G_{ESF'}$ to $G_{ESF'-1}$, $G_{ESF'-1}$ to $G_{ESF'-8}$. And then we prove the indistinguishability of $G_{ESF'-8}$ and $G_{SF''}$.

Indistinguishability of $G_{ESF'}$ and $G_{ESF'-1}$. For the security proof of the indistinguishability of $G_{ESF'}$ and $G_{ESF'-1}$, we define a sequence of games additional hybrid games $G_{F',1}, \ldots, G_{F',h}, \ldots, G_{F',q_n}$, where $G_{ESF'} = G_{F',0}$ and q_n is the number of all node identifiers that are used in HIBE private keys and IBE private keys of an adversary. In the game $G_{F',h}$ for $1 \leq h \leq q_n$, the challenge ciphertext is ESF-1, all IBE private keys are ESF-2, HIBE private keys with a node index $i_n \leq h$ are of ESF-3, the remaining HIBE private keys with a node index $i_n > h$ are ESF-2.

Oracle $O_{3.1}$ This oracle initializes in the same way as O_3 and provides the attacker with initial group elements from the same distribution. Upon receiving a challenge HIBE-key-type query for $I \in Z_n$, it chooses $r, y' \in Z_n$ randomly, and also chooses $X_2, Y_2 \in G_{p_2}$ and $X_3, Y_3 \in G_{p_3}$ randomly. It returns the group elements

$$(w^{y'} g_3^{y'\psi}, g^{y'} g_3^{y'}, v^{y'}(u^{I_j}h)^r X_2 X_3, g^r Y_2 Y_3) \quad (31)$$

to the attacker. It responds to a ciphertext-type query or a challenge IBE-key-type query in the same way as O_3.

We define games $F_1, \cdots, F_{h_c}, \ldots, F_{q_s}$ where $F_0 = G_{S',h}$ and $F_{q_s} = G_{S',h+1}$, and q_s is the maximun number of HIBE private key queries for the node index h. The games are formally defined as follows:

Game F_{h_c} This game F_{h_c} for $1 \leq h_c \leq q_s$ is almost the same as $G_{F',h}$ except the generation of HIBE private keys and IBE private keys with the node index h. An IBE private key with an index pair (h, i_c) is generated as ESF-2. An HIBE private key with an index pair (h, i_c) is generated as follows:

1. $i_c \leq h_c$: It generates a ESF-3 $SK_{HIBE,h}$ by using the element groups in Eq 31.
2. $i_c > h_c$: It simply generates a ESF-2 HIBE private key.

Lemma 8 *Under Assumptions 3, no PPT attacker can distinguish between O_3 and $O_{3.1}$ with non-negligible advantage. So no PPT attacker can distinguish between F_{h_c-1} and F_{h_c} with non-negligible advantage.*

Let $Adv_{\mathcal{A}}^{G_{F',h}}$ be the advantage of $\mathcal{A}$ in a game $G_{F',h}$. From the Lemma 8, we obtain the following equation

$$Adv_{\mathcal{A}}^{G_{ESF'}} - Adv_{\mathcal{A}}^{G_{ESF'-1}} \leq \sum_{h=1}^{q_n} (Adv_{\mathcal{A}}^{G_{F',h-1}} - Adv_{\mathcal{A}}^{G_{F',h}}) \leq O(q_n q_s)(Adv_{\mathcal{B}}^{A3}) \tag{32}$$

Indistinguishability of $G_{ESF'-1}$ and $G_{ESF'-2}$. For the security proof of the indistinguishability of $G_{ESF'-1}$ and $G_{ESF'-2}$, we define a sequence of games $G_{F'-1,1}, \ldots, G_{F'-1,h}, \ldots, G_{F'-1,q_n}$, where $G_{ESF'-1} = G_{F'-1,0}$ and q_n is the number of all node identifiers that are used in HIBE private keys and IBE private keys of an adversary. In the game $G_{F'-1,h}$ for $1 \leq h \leq q_n$, the challenge ciphertext is ESF-1, all HIBE private keys are ESF-3, IBE private keys with a node index $i_n \leq h$ are of ESF-3, the remaining HIBE private keys with a node index $i_n > h$ are ESF-2.

Oracle $O_{3.2}$ This oracle initializes in the same way as $O_{3.1}$ and provides the attacker with initial group elements from the same distribution. Upon receiving a challenge IBE-key-type query for $T \in Z_n$, it chooses $r, y' \in Z_n$ randomly, and also chooses $X_2, Y_2 \in G_{p_2}$ and $X_3, Y_3 \in G_{p_3}$ randomly. It returns the group elements

$$(w_0^{y'} g_3^{y'\psi}, g^{y'} g_3^{y'}, v_0^{y'} (u_0^T h_0)^r X_2 X_3, g^r Y_2 Y_3) \tag{33}$$

to the attacker. It responds to a ciphertext-type query or a challenge HIBE-key-type query in the same way as $O_{3.1}$.

We define hybrid games $F1_1, \cdots, F1_{h_c}, \ldots, F1_{q_e}$ where $F1_0 = G_{F'-1,h}$ and $F1_{q_e} = G_{F'-1,h+1}$, and q_e is the maximun number of IBE private key queries for the node index h. The games are formally defined as follows:

Game $F1_{h_c}$ This game $F1_{h_c}$ for $1 \leq h_c \leq q_s$ is almost the same as $G_{F'-1,h}$ except the generation of HIBE private keys and IBE private keys with the node index h. A HIBE private key with an index pair (h, i_c) is generated as ESF-3. An IBE private key with an index pair (h, i_c) is generated as follows:

1. $i_c \leq h_c$: It generates a ESF-3 SK_{IBE,h_c}.
2. $i_c > h_c$: It simply generates a ESF-2 IBE private key.

Lemma 9 *Under Assumptions 3, no PPT attacker can distinguish between $O_{3.1}$ and $O_{3.2}$ with non-negligible advantage. So no PPT attacker can distinguish between $F1_{h_c-1}$ and $F1_{h_c}$ with non-negligible advantage.*

Let $Adv_{\mathcal{A}}^{G_{F'-1,h}}$ be the advantage of $\mathcal{A}$ in a game $G_{F'-1,h}$. From the Lemma 9, we obtain the following equation

$$Adv_{\mathcal{A}}^{G_{ESF'-1}} - Adv_{\mathcal{A}}^{G_{ESF'-2}} \leq \sum_{h=1}^{q_n} (Adv_{\mathcal{A}}^{G_{F'-1,h-1}} - Adv_{\mathcal{A}}^{G_{F'-1,h}}) \leq O(q_n q_e)(Adv_{\mathcal{B}}^{A3}) \tag{34}$$

Indistinguishability of $G_{ESF'-2}$ and $G_{ESF'-3}$. For the security proof of the indistinguishability of $G_{ESF'-2}$ and $G_{ESF'-3}$, we define the oracle below.

Oracle $O_{3.3}$ This oracle initializes a bit differently from the other oracles. It fixes random elements $g, u, h, v, w, u_0, h_0, v_0, w_0 \in G_{p_1}, g_2 \in G_{p_2}, g_3 \in G_{p_3}$. It chooses random exponents $s, \gamma, \delta_1, \delta_2, y, y_0, \psi, \sigma_1, \sigma_2, a', b', t_3, t_3', t_3'' \in Z_N$. It initially provides the attacker with the group elements:

$$\begin{gathered}(g, u, h, v, w, g^s(g_2g_3)^{\gamma}, w^y(g_2g_3)^{y\psi}, g^y(g_2g_3)^{y}, v^y(g_2g_3)^{y\sigma_1}, \\ u_0, h_0, v_0, w_0, w_0^{y_0}(g_2g_3)^{y_0\psi}, g^{y_0}(g_2g_3)^{y_0}, v_0^{y_0}(g_2g_3)^{y_0\sigma_2})\end{gathered} \tag{35}$$

What differs from the previous oracles here is the added g_3^{γ} and term: notice that this is uniformly random in G_{p_3}, since γ is random modulo p_3 (and uncorrelated from its value modulo p_2). This oracle answers the challenge-key type query in the same way as $O_{3.2}$. To answer a ciphertext-type query for I, it chooses random values $t \in Z_N$ and responds with:

$$(w^s g_2^{\delta_1} v^t g_2^{\sigma_1 t} g_3^{t_3}, g^t g_2^t g_3^{t_3'}, (u^I h)^t g_2^{t(a'I+b')} g_3^{t_3''}) \tag{36}$$

To answer a ciphertext-type query for T, it chooses random values $t \in Z_N$ and responds with:

$$(w_0^s g_2^{\delta_2} v_0^t g_2^{\sigma_2 t} g_3^{t_3}, g^t g_2^t g_3^{t_3'}, (u_0^T h_0)^t g_2^{t(a'T+b')} g_3^{t_3''}) \tag{37}$$

It is crucial to note that these G_{p_3} terms arethe same for each ciphertext-type query response.

Lemma 10 *Under Assumptions 4, no PPT attacker can distinguish between $O_{3.2}$ and $O_{3.3}$ with non-negligible advantage. So no PPT attacker can distinguish between $G_{ESF'-2}$ and $G_{ESF'-3}$ with non-negligible advantage.*

From the Lemma 10, we obtain the following equation

$$Adv_{\mathcal{A}}^{G_{ESF'-2}} - Adv_{\mathcal{A}}^{G_{ESF'-3}} \leq (Adv_{\mathcal{B}}^{A4}) \tag{38}$$

Indistinguishability of $G_{ESF'-3}$ and $G_{ESF'-4}$: For the security proof of the indistinguishability of $G_{ESF'-3}$ and $G_{ESF'-4}$, we define a sequence of games additional hybrid games $G_{F'-3,1}, \ldots, G_{F'-3,h}, \ldots, G_{F'-3,q_n}$, where $G_{ESF'-3} = G_{F'-3,0}$ and q_n is the number of all node identifiers that are used in HIBE private keys and IBE private keys of an adversary. In the game $G_{F'-3,h}$ for $1 \leq h \leq q_n$, the challenge ciphertext is ESF-5, all IBE private keys are ESF-3, HIBE private keys with a node index $i_n \leq h$ are of ESF-4, the remaining HIBE private keys with a node index $i_n > h$ are ESF-3.

Oracle $O_{3.4}$ This oracle initializes in the same way with $O_{3.3}$ and provides the attacker the same initial elements as $O_{3.3}$. This oracle answers the ciphertext-type query and IBE key- type query in the same way as $O_{3.3}$. To answer a challenge HIBE private key type query for I, it chooses random values $y, r \in Z_N, X_2, Y_2 \in G_{p_2}$ randomly, and $X_3, Y_3 \in G_{p_3}$ and responds with:

$$(w^{y'}(g_2g_3)^{y'\psi}, g^{y'}(g_2g_3)^{y'}, v^{y'}(u^I h)^r X_2X_3, g^r Y_2Y_3) \tag{39}$$

We define hybrid games $F3_1, \cdots, F3_{h_c}, \ldots, F3_{q_s}$ where $F3_0 = G_{F'-3,h}$ and $F3_{q_s} = G_{F'-3,h+1}$, and q_s is the maximun number of HIBE private key queries for the node index h. The games are formally defined as follows:

Game $F3_{h_c}$ This game $F3_{h_c}$ for $1 \leq h_c \leq q_s$ is almost the same as $G_{F'-3,h}$ except the generation of HIBE private keys and IBE private keys with the node index h. An IBE private key with an index pair (h, i_c) is generated as ESF-3. An HIBE private key with an index pair (h, i_c) is generated as follows:

1. $i_c \leq h_c$: It generates a ESF-4 $SK_{HIBE,h}$ by using the element groups in Eq 39.
2. $i_c > h_c$: It simply generates a ESF-3 HIBE private key.

Lemma 11 *Under Assumptions 4, no PPT attacker can distinguish between* $O_{3.3}$ *and* $O_{3.4}$ *with non-negligible advantage. So no PPT attacker can distinguish between* $G_{ESF'-3}$ *and* $G_{ESF'-4}$ *with non-negligible advantage.*

Let $Adv_{\mathcal{A}}^{G_{F'-3,h}}$ be the advantage of $\mathcal{A}$ in a game $G_{F'-3,h}$. From the Lemma 11, we obtain the following equation

$$Adv_{\mathcal{A}}^{G_{ESF'-3}} - Adv_{\mathcal{A}}^{G_{ESF'-4}} \leq \sum_{h=1}^{q_n} (Adv_{\mathcal{A}}^{G_{F'-3,h-1}} - Adv_{\mathcal{A}}^{G_{F'-3,h}}) \leq O(q_n q_s)(Adv_{\mathcal{B}}^{A4}) \tag{40}$$

Indistinguishability of $G_{ESF'-4}$ **and** $G_{ESF'-5}$. For the security proof of the indistinguishability of $G_{ESF'-4}$ and $G_{ESF'-5}$, we define a sequence of games additional hybrid games $G_{F'-4,1}, \ldots, G_{F'-4,h}, \ldots, G_{F'-4,q_n}$, where $G_{ESF'-4} = G_{F'-4,0}$ and q_n is the number of all node identifiers that are used in HIBE private keys and IBE private keys of an adversary. In the game $G_{F'-4,h}$ for $1 \leq h \leq q_n$, the challenge ciphertext is ESF-5, all HIBE private keys are ESF-4, IBE private keys with a node index $i_n \leq h$ are of ESF-4, the remaining IBE private keys with a node index $i_n > h$ are ESF-3.

Oracle $O_{3.5}$ This oracle initializes in the same way with $O_{3.4}$ and provides the attacker the same initial elements as $O_{3.4}$. This oracle answers the ciphertext-type query and HIBE key-type query in the same way as $O_{3.4}$. To answer a challenge IBE private key type query for T, it chooses random values $y, r \in Z_N$, $X_2, Y_2 \in G_{p_2}$ randomly, and $X_3, Y_3 \in G_{p_3}$ and responds with:

$$(w_0^{y'}(g_2 g_3)^{y'\psi}, g^{y'}(g_2 g_3)^{y'}, v_0^{y'}(u_0^T h_0)^r X_2 X_3, g^r Y_2 Y_3) \tag{41}$$

We define hybrid games $F4_1, \cdots, F4_{h_c}, \ldots, F4_{q_s}$ where $F4_0 = G_{F'-4,h}$ and $F4_{q_e} = G_{F'-4,h+1}$, and q_e is the maximun number of IBE private key queries for the node index h. The games are formally defined as follows:

Game $F4_{h_c}$ This game $F4_{h_c}$ for $1 \leq h_c \leq q_e$ is almost the same as $G_{F'-4,h}$ except the generation of HIBE private keys and IBE private keys with the node index h. An HIBE private key with an index pair (h, i_c) is generated as ESF-4. An IBE private key with an index pair (h, i_c) is generated as follows:

1. $i_c \leq h_c$: It generates a ESF-4 $SK_{IBE,h}$ by using the element groups in Eq 41.
2. $i_c > h_c$: It simply generates a ESF-3 IBE private key.

Lemma 12 *Under Assumptions 4, no PPT attacker can distinguish between* $O_{3.4}$ *and* $O_{3.5}$ *with non-negligible advantage. So no PPT attacker can distinguish between* $G_{ESF'-4}$ *and* $G_{ESF'-5}$ *with non-negligible advantage.*

Let $Adv_{\mathcal{A}}^{G_{F'-4,h}}$ be the advantage of $\mathcal{A}$ in a game $G_{F'-4,h}$. From the Lemma 12, we obtain the following equation

$$Adv_{\mathcal{A}}^{G_{ESF'-4}} - Adv_{\mathcal{A}}^{G_{ESF'-5}} \leq \sum_{h=1}^{q_n} (Adv_{\mathcal{A}}^{G_{F'-4,h-1}} - Adv_{\mathcal{A}}^{G_{F'-4,h}}) \leq O(q_n q_e)(Adv_{\mathcal{B}}^{A4}) \tag{42}$$

Indistinguishability of $G_{ESF'-5}$ and $G_{ESF'-6}$. For the security proof of the indistinguishability of $G_{ESF'-5}$ and $G_{ESF'-6}$, we define the oracle below.

Oracle $O_{3.6}$ This oracle fixes random elements $g, u, h, v, w, u_0, h_0, v_0, w_0 \in G_{p_1}, g_2 \in G_{p_2}$, $g_3 \in G_{p_3}$. It chooses random exponents $s, \gamma, \delta_1, \delta_2, y, y_0, \psi, \sigma_1, \sigma_2, a', b', t_3, t_3', t_3'' \in Z_N$. It initially provides the attacker with the group elements:

$$\begin{gathered}(g, u, h, v, w, g^s(g_2g_3)^{\gamma}, w^y(g_2g_3)^{y\psi}, g^y(g_2g_3)^{y}, v^y(g_2g_3)^{y\sigma_1}, \\ u_0, h_0, v_0, w_0, w_0^{y_0}(g_2g_3)^{y_0\psi}, g^{y_0}(g_2g_3)^{y_0}, v_0^{y_0}(g_2g_3)^{y_0\sigma_2})\end{gathered} \tag{43}$$

What differs from the previous oracles here is the added g_3^{γ} and term: notice that this is uniformly random in G_{p_3}, since γ is random modulo p_3 (and uncorrelated from its value modulo p_2). This oracle answers the challenge-key type query in the same way as $O_{3.2}$. To answer a ciphertext-type query for I, it chooses random values $t \in Z_N$ and responds with:

$$(w^s g_2^{\delta_1} v^t g_2^{\sigma_1 t} g_3^{t_3}, g^t g_2^t g_3^{t'}, (u^I h)^t g_2^{t(a'I+b')} g_3^{t''}) \tag{44}$$

To answer a ciphertext-type query for T, it chooses random values $t \in Z_N$ and responds with:

$$(w_0^s g_2^{\delta_2} v_0^t g_2^{\sigma_2 t} g_3^{t_3}, g^t g_2^t g_3^{t'}, (u_0^T h_0)^t g_2^{t(a'T+b')} g_3^{t''}) \tag{45}$$

It is crucial to note that these G_{p_3} terms arethe same for each ciphertext-type query response.

Lemma 13 *Under Assumptions 4, no PPT attacker can distinguish between $O_{3.5}$ and $O_{3.6}$ with non-negligible advantage. So no PPT attacker can distinguish between $G_{ESF'-5}$ and $G_{ESF'-6}$ with non-negligible advantage.*

From the Lemma 13, we obtain the following equation

$$Adv_{\mathcal{A}}^{G_{ESF'-5}} - Adv_{\mathcal{A}}^{G_{ESF'-6}} \leq (Adv_{\mathcal{B}}^{A4}) \tag{46}$$

Indistinguishability of $G_{ESF'-6}$ and $G_{ESF'-7}$. For the security proof of the indistinguishability of $G_{ESF'-6}$ and $G_{ESF'-7}$, we define a sequence of games $G_{F'-6,1}, \ldots, G_{F'-6,h}, \ldots, G_{F'-6,q_n}$, where $G_{ESF'-6} = G_{F'-6,0}$ and q_n is the number of all node identifiers that are used in HIBE private keys and IBE private keys of an adversary. In the game $G_{F'-6,h}$ for $1 \leq h \leq q_n$, the challenge ciphertext is ESF-1, all HIBE private keys are ESF-4, IBE private keys with a node index $i_n \leq h$ are semi-functional, the remaining IBE private keys with a node index $i_n > h$ are ESF-4.

Oracle $O_{7/2'}$ This oracle initializes in the same way as $O_{3.6}$ and provides the attacker with initial group elements from the same distribution. Upon receiving a challenge IBE-key-type query for $T \in Z_n$, it chooses $r, y' \in Z_n$ randomly, and also chooses $X_2, Y_2 \in G_{p_2}$ and $X_3, Y_3 \in G_{p_3}$ randomly. It returns the group elements

$$(w_0^{y'}(g_2g_3)^{y'\psi}, g^{y'}(g_2g_3)^{y'}, v_0^{y'}(u_0^T h_0)^r X_2 X_3, g^r Y_3) \tag{47}$$

to the attacker. It responds to a ciphertext-type query or a challenge HIBE-key-type query in the same way as $O_{3.6}$.

We define hybrid games $F6_{1,1}$, $F6_{1,2}$, $\cdots$, $F6_{h_c,1}$, $F6_{h_c,2}$, $\ldots$, $F6_{q_s,1}$, $F6_{q_s,2}$ where $F6_{0,2} = G_{F'-6,h}$ and $F6_{q_e,2} = G_{F'-6,h+1}$, and q_e is the maximun number of IBE private key queries for the node index h. The games are formally defined as follows:

Game $F6_{h_c,1}$ This game $F6_{h_c,1}$ for $1 \leq h_c \leq q_e$ is almost the same as $G_{F'-6,h}$ except the generation of HIBE private keys and IBE private keys with the node index h. An HIBE private key with an index pair (h, i_c) is generated as ESF-4. An IBE private key with an index pair (h, i_c) is generated as follows:

1. $i_c < h_c$: It generates a semi-functional key $SK_{IBE,h}$.
2. $i_c = h_c$: It generates a normal $SK'_{IBE,h}$ and converts the key to a ESF-5 $SK_{IBE,h}$ by using the element groups in Eq 47.
3. $i_c > h_c$: It generates a ESF-4 IBE private key.

Game $F6_{h_c,2}$ This game $F6h_c$, 2 for $1 \leq h_c \leq q_e$ is almost the same as $F6_{h_c,1}$ except the generation of IBE private key with an index pair (h, i_c) and $i_c = h_c$ is generated as a semi-functional $SK_{IBE,h}$.

Lemma 14 *Under Assumptions 4, no PPT attacker can distinguish between* $O_{3.6}$ *and* $O_{7/2'}$ *with non-negligible advantage. So no PPT attacker can distinguish between* $F6_{i-1,2}$ *and* $F6_{i,1}$ *with non-negligible advantage.*

Oracle $\widetilde{O}_i^*$ This oracle initializes in the same way as O_i^*, and provides the attacker with initial group elements from the same distribution. It also responds to a ciphertext-type query as same as O_i^*. It responds to a HIBE-key-type query in the same way as $O_{3.6}$. Upon receiving a challenge IBE-key-type query for $T \in Z_n$, it chooses $r, y' \in Z_n$ randomly, and returns the group elements

$$(w_0^{y'}(g_2g_3)^{y'\psi}, g^{y'}(g_2g_3)^{y'}, v_0^{y'}(g_2g_3)^{y'\sigma_2}(u_0^T h_0)^r, g^r(g_2g_3)^{y'}) \tag{48}$$

to the attacker.

Lemma 15 *Under Assumptions 3, no PPT attacker can distinguish between* $O_{7/2'}$ *and* $\widetilde{O}_{q_c}^*$ *with non-negligible advantage. So no PPT attacker can distinguish between* $F6_{i,1}$ *and* $F6_{i,2}$ *with non-negligible advantage.*

Let $Adv_{\mathcal{A}}^{G_{E',h}}$ be the advantage of $\mathcal{A}$ in a game $G_{E',h}$. From the Lemma 14, 15, we obtain the following equation

$$\begin{aligned} Adv_{\mathcal{A}}^{G_{F'-6,h-1}} - Adv_{\mathcal{A}}^{G_{F'-6,h}} &\leq \textstyle\sum_{h_c=1}^{q_e}(|Adv_{\mathcal{A}}^{F6_{h_c,1}} - Adv_{\mathcal{A}}^{F6_{h_c,2}}| + |Adv_{\mathcal{A}}^{F6_{h_c-1,2}} - Adv_{\mathcal{A}}^{F6_{h_c,1}}|) \\ &\leq O(q_e)(Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4}) \end{aligned}$$

So we obtain the following equation

$$Adv_{\mathcal{A}}^{G_{ESF'-6}} - Adv_{\mathcal{A}}^{G_{ESF'-7}} \leq \sum_{h=1}^{q_n}(Adv_{\mathcal{A}}^{G_{E',h-1}} - Adv_{\mathcal{A}}^{G_{E',h}}) \leq O(q_n q_e)(Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4}) \tag{49}$$

Indistinguishability of $G_{ESF'-7}$ **and** $G_{ESF'-8}$. We now prove the indistinguishability of $G_{ESF'-7}$ and $G_{ESF'-8}$ in a hybrid argument using polynomially many steps. We let q_c denote the number of ciphertext-type queries made by a PPT attacker $\mathcal{A}$. Firstly we define hybrid games $S'_{-1,1}, S'_{0,2}, S'_{0,3}, S'_{0,1}, S'_{1,2}, S'_{1,3}, S'_{1,1} \ldots, S'_{k,2}, S'_{k,3}, S'_{k,1}, \ldots, S'_{q_c-1,2}, S'_{q_c-1,3}, S'_{q_c-1,1}$, where $S'_{-1,1} = G_{ESF'-7}$ and $S'_{q_c-1,1} = G_{ESF'-8}$. The games are formally defined as follows:

Game $S'_{k,1}$ This game $S'_{k,1}$ for $0 \leq k \leq q_c$ is almost the same as G_{ESF-7} except the generation of the challenge ciphertext. The challenge ciphertext of $(T^*, I_1^*, \ldots, I_{q_c-1}^*)$ is generated as EST-2^k-CT outputed by **EncryptESF-2**^{k}.

Game $S'_{k,2}$ This game $S'_{k,2}$ for $0 \leq k \leq q_c - 1$ is almost the same as G_{ESF-7} except the generation of the challenge ciphertext. The challenge ciphertext of $(T^*, I_1^*, \ldots, I_{q_c-1}^*)$ is generated as EST-3^k-CT outputed by **EncryptESF-3**^{k}.

Game $S'_{k,3}$ This game $S'_{k,3}$ for $0 \leq k \leq q_c - 1$ is almost the same as G_{ESF-7} except the generation of the challenge ciphertext. The challenge ciphertext of $(T^*, I_1^*, \ldots, I_{q_c-1}^*)$ is generated as EST-4^k-CT outputed by **EncryptESF-4**^{k}.

We will define additional oracles $\widetilde{O}_i^*$ for each i from 0 to $q_c - 1$, $\widetilde{O}'_i$ for each i from 0 to $q_c - 1$, and $\widetilde{O}''_i$ for each i from 0 to $q_c - 1$.

Oracle $\widetilde{O}'_i$ This oracle acts the same as $\widetilde{O}_i^*$ except that its response to the ciphertext-type query is as same as O'_i.

Oracle $\widetilde{O}''_i$ This oracle acts the same as $\widetilde{O}_i^*$ except that its response to the ciphertext-type query is as same as O''_i.

Lemma 16 *Under Assumptions 3, no PPT attacker can distinguish between* $\widetilde{O}_k^*$ *and* $\widetilde{O}''_k$ *with non-negligible advantage. So no PPT attacker can distinguish between* $S'_{k,1}$ *and* $S'_{k,3}$ *with non-negligible advantage.*

Lemma 17 *Under Assumptions 4, no PPT attacker can distinguish between* $\widetilde{O}''_k$ *and* $\widetilde{O}'_k$ *with non-negligible advantage. So no PPT attacker can distinguish between* $S'_{k,3}$ *and* $S'_{k,2}$ *with non-negligible advantage.*

Lemma 18 *Under Assumptions 3, no PPT attacker can distinguish between* $\widetilde{O}'_k$ *and* $\widetilde{O}_{k-1}^*$ *with non-negligible advantage. So no PPT attacker can distinguish between* $S'_{k,2}$ *and* $S'_{k-1,1}$ *with non-negligible advantage.*

Let $Adv_{\mathcal{A}}^{S'_{k,1}}$, $Adv_{\mathcal{A}}^{S'_{k,2}}$ and $Adv_{\mathcal{A}}^{S'_{k,3}}$ be the advantage of $\mathcal{A}$ in the games $S'_{k,1}$, $S'_{k,2}$ and $S'_{k,3}$. From the Lemma 16, 17, 18, we obtain the following equation

$$\begin{aligned} Adv_{\mathcal{A}}^{G_{ESF'-7}} - Adv_{\mathcal{A}}^{G_{ESF'-8}} &\leq \textstyle\sum_{k=0}^{q_c-1} (|Adv_{\mathcal{A}}^{S'_{k-1,1}} - Adv_{\mathcal{A}}^{S'_{k,2}}| + |Adv_{\mathcal{A}}^{S'_{k,2}} - Adv_{\mathcal{A}}^{S'_{k,3}}| \\ &\quad + |Adv_{\mathcal{A}}^{S'_{k,3}} - Adv_{\mathcal{A}}^{S'_{k,1}}|) \\ &\leq q_c(2Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4}) \end{aligned} \tag{50}$$

Indistinguishability of $G_{ESF'-8}$ **and** $G_{SF''}$. For the security proof of the indistinguishability of $G_{ESF'-8}$ and $G_{SF''}$, we define a sequence of games $G_{F'-8,1}, \ldots, G_{F'-8,h}, \ldots, G_{F'-8,q_n}$, where $G_{ESF'-8} = G_{F'-8,0}$ and q_n is the number of all node identifiers that are used in HIBE private keys and IBE private keys of an adversary. In the game $G_{F'-8,h}$ for $1 \leq h \leq q_n$, the challenge ciphertext is semi-functional, all IBE private keys are semi-functional, HIBE private keys with a node index $i_n \leq h$ are semi-functional, the remaining HIBE private keys with a node index $i_n > h$ are ESF-4.

Oracle $O_{7/2}$ This oracle initializes in the same way as $\widetilde{O}_0^*$ and provides the attacker with initial group elements from the same distribution. Upon receiving a challenge HIBE-key-type query for $I \in Z_n$, it chooses $r, y' \in Z_n$ randomly, and also chooses X_2, $Y_2 \in G_{p_2}$ and X_3, $Y_3 \in G_{p_3}$ randomly. It returns the group elements

$$(w^{y'}(g_2g_3)^{y'\psi}, g^{y'}(g_2g_3)^{y'}, v^{y'}(u^I h)^r X_2X_3, g^r Y_3) \tag{51}$$

to the attacker. It responds to a ciphertext-type query or a challenge IBE-key-type query in the same way as $\widetilde{O}_0^*$.

We define hybrid games $I_{1,1}, I_{1,2}, \cdots, I_{h_c,1}, I_{h_c,2}, \ldots, I_{q_s,1}, I_{q_s,2}$ where $I_{0,2} = G_{F'-8,h}$ and $I_{q_s,2} = G_{F'-8,\, h+1}$, and q_s is the maximun number of HIBE private key queries for the node index h. The games are formally defined as follows:

Game $I_{h_c,1}$ This game $I_{h_c,1}$ for $1 \leq h_c \leq q_s$ is almost the same as $G_{F'-8,h}$ except the generation of HIBE private keys and IBE private keys with the node index h. An IBE private key with an index pair (h, i_c) is generated as a semi-functional key. An HIBE private key with an index pair (h, i_c) is generated as follows:

1. $i_c < h_c$: It generates a semi-functional $SK_{HIBE,h}$.
2. $i_c = h_c$: It generates a ESF-5 $SK_{HIBE,h}$.
3. $i_c > h_c$: It generates a ESF-4 $SK_{HIBE,h}$.

Game $H_{h_c,2}$ This game $H_{h_c,1}$ for $1 \leq h_c \leq q_s$ is almost the same as $H_{h_c,1}$ except the generation of HIBE private key with an index pair (h, i_c) and $i_c = h_c$ is generated as a semi-functional $SK_{HIBE,h}$.

Lemma 19 *Under Assumptions 4, no PPT attacker can distinguish between* $\widetilde{O}_0^*$ *and* $O_{7/2}$ *with non-negligible advantage. So no PPT attacker can distinguish between* $I_{h_c-1,2}$ *and* $I_{h_c,1}$ *with non-negligible advantage.*

Lemma 20 *Under Assumptions 3, no PPT attacker can distinguish between* $O_{7/2}$ *and* O_4 *with non-negligible advantage. So no PPT attacker can distinguish between* $I_{h_c,1}$ *and* $I_{h_c,2}$ *with non-negligible advantage.*

Let $Adv_{\mathcal{A}}^{G_{E',h}}$ be the advantage of $\mathcal{A}$ in a game $G_{E',h}$. From the Lemma 19, 20, we obtain the following equations

$$\begin{aligned} Adv_{\mathcal{A}}^{G_{F'-8,h-1}} - Adv_{\mathcal{A}}^{G_{F'-8,h}} &\leq \textstyle\sum_{h_c=1}^{q_s}(|Adv_{\mathcal{A}}^{I_{h_c,1}} - Adv_{\mathcal{A}}^{I_{h_c,2}}| + |Adv_{\mathcal{A}}^{I_{h_c-1,2}} - Adv_{\mathcal{A}}^{I_{h_c,1}}|) \\ &\leq O(q_s)(Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4}) \end{aligned}$$

and

$$Adv_{\mathcal{A}}^{G_{ESF'-8}} - Adv_{\mathcal{A}}^{G_{SF''}} \leq \sum_{h=1}^{q_n}(Adv_{\mathcal{A}}^{G_{F'-8,h-1}} - Adv_{\mathcal{A}}^{G_{F'-8,h}}) \leq O(q_n q_s)(Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4}) \tag{52}$$

So we obtain the following equation

$$\begin{aligned} & Adv_{\mathcal{A}}^{G_{ESF'}} - Adv_{\mathcal{A}}^{G_{SF''}} \leq |(Adv_{\mathcal{A}}^{G_{ESF'}} - Adv_{\mathcal{A}}^{G_{ESF'-1}}| + \\ & \qquad \textstyle\sum_{i=1}^{7} |(Adv_{\mathcal{A}}^{G_{ESF'-i}} - Adv_{\mathcal{A}}^{G_{ESF'-i+1}}| + |(Adv_{\mathcal{A}}^{G_{ESF'-8}} - Adv_{\mathcal{A}}^{G_{SF''}}| \\ & \leq O(q_n q_s)(Adv_{\mathcal{B}}^{A3}) + O(q_n q_e)(Adv_{\mathcal{B}}^{A3}) + (Adv_{\mathcal{B}}^{A4}) + O(q_n q_s)(Adv_{\mathcal{B}}^{A4}) + \\ & O(q_n q_e)(Adv_{\mathcal{B}}^{A4}) + (Adv_{\mathcal{B}}^{A4}) + O(q_n q_e)(Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4}) + \\ & O(q_c)(2Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4}) + O(q_n q_s)(Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4}) \\ & \leq (O(q_n(q_s + q_e)) + O(l))(Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4}) \end{aligned} \tag{53}$$

According to the equations Eqs 23, 28, 30, 53, we obtain the following equation

$$\begin{aligned} Adv_{\mathcal{A}}^{G_{C'}} - Adv_{\mathcal{A}}^{G_{SF''}} &\leq |(Adv_{\mathcal{A}}^{G_{C'}} - Adv_{\mathcal{A}}^{G_{E-S}}| + |(Adv_{\mathcal{A}}^{G_{E-S}} - Adv_{\mathcal{A}}^{G_{E-S'}}| \\ &+ |(Adv_{\mathcal{A}}^{G_{E-S'}} - Adv_{\mathcal{A}}^{G_{ESF'}}| + |(Adv_{\mathcal{A}}^{G_{ESF'}} - Adv_{\mathcal{A}}^{G_{SF''}}| \\ &\leq (O(q_n(q_s + q_e)) + O(l))(Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4}) \\ &\leq (O(q_n log N_{max}) + O(q_n r_{max} log N_{max}) + O(l))(Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4}) \end{aligned} \tag{54}$$

4.4.3 Type-2 adversary. The Type-2 adversary is restricted to queries on the update keys on the time $T \notin T^*$. So we could show an information theoretic argument for the update keys and avoid a potential paradox for the secret keys, in the similar way of the situation of the Type-1 adversary in Sec.4.4.2.

The proof strategy for the indistinguishabilities between games $G_{C'}$ to G_{EK-U}, and to G_{ESF} under the Type-2 adversary is by going through several intermediary oracles in Table 6, where the type settings of the update keys and the secret keys in every oracle and game respectively are swaped compared to the setting in Sec.4.4.2. The proof of every respective lemma is similar to the proof for the Type-1 adversary, and finally we obtain the advantage between $G_{C'}$ and $G_{ESF'}$ under the Type-2 adversary as same in Eq 54.

4.5 Indistinguishability of $G_{SF''}$ and $G_{SF'}$

In the game $G_{SF''}$, the type of ciphertexts, secret keys and update keys are all semi-functional, except the decryption keys are normal. In this section, we give the proof of the indistinguishability of $G_{SF''}$ and $G_{SF'}$ via a hybrid argument over the sequence of games $G_{SF''}$, G_{E-D}, G_{ESF} and $G_{SF'}$ to transform the type of decryption keys from normal to ephemeral semi-functional, and then to semi-functional.

The hybrid argument we conduct for the indistinguishability of $G_{SF''}$ and $G_{SF'}$ is following the process similar to the argument for the indistinguishability of $G_{C'}$ and $G_{SF''}$. But it is simpler since the transformation of challenge type only happens to the decryption keys and the challenge ciphertexts. So we just treat the decryption keys as a secret key of the identity $(T, id_1, \cdots, id_j)$ and follow the proof strategy in the nested dual system encryption of the unbounded HIBE [12].

We show the oracles for proving the the indistinguishability of $G_{SF''}$ and $G_{SF'}$ in Table 7 which answer queries from the challenger $\mathcal{B}$ by sampling various distributions of group elements to construct the decryption keys, challenge ciphertexts and also the secret keys and update keys.

Since these oracles initially provide the attacker with a description of the group G, as well as the group elements

$$g, u, v, w, g^s g_2^{\gamma}, w^y (g_2 g_3)^{\gamma\psi}, g^y (g_2 g_3)^y, v^y (g_2 g_3)^{y\sigma},$$

$$u_0, v_0, w_0, w_0^{y_0} (g_2 g_3)^{y_0\psi}, g^{y_0} (g_2 g_3)^{y_0}, v_0^{y_0} (g_2 g_3)^{y_0\sigma}$$

So the simulation of semi-functional secret keys and update keys are achievable in all oracles and games.

(1) Indistinguishability of $G_{SF''}$ and G_{E-D}

For the security proof of the indistinguishability of $G_{SF''}$ and G_{E-D}, we define a sequence of additional hybrid games $J_{1,1}, J_{1,2}, \cdots, J_{h_d,1}, J_{h_d,2}, \ldots, J_{q_d,1}, J_{q_d,2}$ where $J_{0,2} = G_{SF''}$ and $J_{q_d,2} = G_{E-D}$,

Table 7. Simulation of challenge keys and cipertext in oracles for the proof of the indistinguishability between $G_{SF'}$ and G_{SF}.

Oracle	CT-Type Response	SK-Type Response	UK-Type Response	DK-Type Response
$\boldsymbol{O_4}$	SF	SF	SF	Normal
$O_{9/2}$	SF	SF	SF	ESF-1
$\boldsymbol{O_5}$	SF	SF	SF	ESF-2
$\grave{O}_i^*$	ESF-2^i	SF	SF	ESF-2
$\grave{O}_i'$	ESF-3^i	SF	SF	ESF-2
$\grave{O}_i''$	ESF-4^i	SF	SF	ESF-2
$\boldsymbol{O_6}$	ESF-1	SF	SF	ESF-2
$O_{6.1}$	ESF-1	SF	SF	ESF-3
†$O_{6.2}$	ESF-5	SF	SF	ESF-3
†$O_{6.3}$	ESF-5	SF	SF	ESF-4
$\acute{O}_i^*$	ESF-2^i	SF	SF	ESF-4
$\acute{O}_i'$	ESF-3^i	SF	SF	ESF-4
$\acute{O}_i''$	ESF-4^i	SF	SF	ESF-4
$O_{13/2}$	SF	SF	SF	ESF-5
$\boldsymbol{O_7}$	SF	SF	SF	SF

Note: oracles marked with † initialize with an extra G_{p_3} term on $g^s g_2^y$.

https://doi.org/10.1371/journal.pone.0195204.t007

and q_d is the number of decryption key queries of an adversary. The games and a additional oracle $O_{9/2}$ used in the proof are formally defined as follows:

Oracle $O_{9/2}$ This oracle initializes in the same way as O_4, O_5 and provides the attacker with initial group elements from the same distribution. Upon receiving a challenge decryption-key-type query for I, $T \in Z_n$, it chooses $r, y', r', y'' \in Z_n$ randomly, and also chooses $X_2, Y_2, X_2', Y_2' \in G_{p_2}, X_3, Y_3, X_3', Y_3' \in G_{p_3}$ randomly. It returns the group elements

$$(w^{y'}, g^{y'}, v^{y'}(u^I h)^r X_3, g^r Y_3, w_0^{y''}, g^{y''}, v_0^{y''}(u_0^T h_0)^{r'} X_3', g^{r'} Y_3') \tag{55}$$

to the attacker. It responds to a ciphertext-type query and a challenge (H)IBE-key-type query in the same way as O_4.

Game $J_{h_d,1}$ This game $J_{h_d,1}$ for $1 \le h_d \le q_d$ is almost the same as $G_{EF'}$ except the generation of the decryption keys $DK_{ID|_k,T}$.

1. $i_d < h_d$: It generates a normal $DK'_{ID|_k,T}$ and converts the key to a ESF-2 $DK_{ID|_k,T}$ by using the element groups in Eq 17.
2. $i_d = h_d$: It generates a normal $DK'_{ID|_k,T}$ and converts the key to a ESF-1 $DK_{ID|_k,T}$ by using the element groups in Eq 55.
3. $i_d > h_d$: It simply generates a normal decryption key.

Game $J_{h_d,2}$ This game $J_{h_d,1}$ for $1 \le h_d \le q_d$ is almost the same as $J_{h_d,1}$ except the generation of the h_d^{st} decryption key is generated as a ESF-2 $DK_{ID|_k,T}$ by using the element groups in Eq 17. The first h_d decryption keys are generated as ESF-2 and the remaining decryption keys are generated as normal.

Lemma 21 *Under Assumptions 3, no PPT attacker can distinguish between O_4 and $O_{9/2}$ with non-negligible advantage. So no PPT attacker can distinguish between $J_{h_d-1,2}$ and $J_{h_d,1}$ with non-negligible advantage.*

Lemma 22 *Under Assumptions 4, no PPT attacker can distinguish between $O_{9/2}$ and O_5 with non-negligible advantage. So no PPT attacker can distinguish between $J_{h_d,1}$ and $J_{h_d,2}$ with non-negligible advantage.*

From the Lemma 21, 22, we obtain the following equation

$$\begin{aligned} Adv_{\mathcal{A}}^{G_{SF''}} - Adv_{\mathcal{A}}^{G_{E-D}} &\leq \sum_{h_d=1}^{q_d} (|Adv_{\mathcal{A}}^{J_{h_d}-1,2} - Adv_{\mathcal{A}}^{J_{h_d},1}| + |Adv_{\mathcal{A}}^{J_{h_d},1} - Adv_{\mathcal{A}}^{J_{h_d},2}|) \\ &\leq O(q_d)(Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4}) \end{aligned} \tag{56}$$

(2) Indistinguishability of G_{E-D} and G_{ESF}

We now prove the indistinguishability of G_{E-D} and G_{ESF} in a hybrid argument using polynomially many steps. We let q_c denote the number of ciphertext-type queries made by a PPT attacker $\mathcal{A}$. Firstly we define hybrid games $L_{-1,1}$, $L_{0,2}$, $L_{0,3}$, $L_{0,1}$, $L_{1,2}$, $L_{1,3}$, $L_{1,1} \cdots$, $L_{k,2}$, $L_{k,3}$, $L_{k,1}$, …, $L_{q_c-1,2}$, $L_{q_c-1,3}$, $L_{q_c-1,1}$, where $L_{-1,1} = G_{E-D}$ and $L_{q_c-1,1} = G_{ESF}$. The games are formally defined as follows:

Game $L_{k,1}$ This game $L_{k,1}$ for $0 \leq k \leq q_c$ is almost the same as G_{E-D} except the generation of the challenge ciphertext. The challenge ciphertext of $(T^*, I_1^*, \ldots, I_{q_c-1}^*)$ is generated as EST-2^k-CT outputed by **EncryptESF-2**^{k}.

Game $L_{k,2}$ This game $L_{k,2}$ for $0 \leq k \leq q_c - 1$ is almost the same as G_{E-D} except the generation of the challenge ciphertext. The challenge ciphertext of $(T^*, I_1^*, \ldots, I_{q_c-1}^*)$ is generated as EST-3^k-CT outputed by **EncryptESF-3**^{k}.

Game $L_{k,3}$ This game $L_{k,3}$ for $0 \leq k \leq q_c - 1$ is almost the same as G_{E-D} except the generation of the challenge ciphertext. The challenge ciphertext of $(T^*, I_1^*, \ldots, I_{q_c-1}^*)$ is generated as EST-4^k-CT outputed by **EncryptESF-4**^{k}.

We will define additional oracles $\grave{O}_i^*$ for each i from 0 to $q_c - 1$, $\grave{O}_i'$ for each i from 0 to $q_c - 1$, and $\grave{O}_i''$ for each i from 0 to $q_c - 1$.

Oracle $\grave{O}_i^*$ This oracle initializes in the same way as O_5, O_6 and provides the attacker with initial group elements from the same distribution. It also responds to challenge key-type queries in the same way as O_5, O_6. It keeps a counter of ciphertext-type queries which is initially equal to zero. It increments this counter after each response to a ciphertext-type query. In response to the j^{th} ciphertext-type query for some I_j^*, if $j \leq i$, it responds exactly like O_6. If $j > i$, it responds exactly like O_5. In particular, $\grave{O}_0^*$ is identical to O_5 and $\grave{O}_q^*$ is identical to O_6.

Oracle $\grave{O}_i'$ This oracle acts the same as $\grave{O}_i^*$ except in its response to the i^{th} ciphertext-type query. For the i^{th} ciphertext-type query for identity I^*, it chooses a random $t \in Z_N$ and random elements X_3, $Y_3 \in G_{p_3}$ and responds with:

$$(w^s g_2^{\delta_1} v^t X_3^{\sigma_1}, g^t X_3, (u^{I_j^*} h)^t Y_3) \tag{57}$$

If $i = 0$, the i^{th} ciphertext-type query is for time T^*. It chooses a random $t_0 \in Z_N$ and random elements X_3', $Y_3' \in G_{p_3}$ and responds with:

$$(w_0^s g_2^{\delta_2} v_0^{t_0} X_3' \sigma_2, g^{t_0} X_3', (u_0^{T^*} h_0)^{t_0} Y_3') \tag{58}$$

Oracle $\grave{O}_i''$ This oracle acts the same as $\grave{O}_i^*$ except in its response to the ith ciphertext-type query. For the i^{th} ciphertext-type query for identity I^*, it chooses a random $t \in Z_N$ and random elements X_3, $Y_3 \in G_{p_3}$ and responds with:

$$(w^s g_2^{\delta_1} v^t X_3^{\sigma_1}, g^t g_2^t X_3, (u^{I_j^*} h)^t g_2^{t(a' I_j^* + b')} Y_3) \tag{59}$$

If $i = 0$, the i^{th} ciphertext-type query is for time T^*. It chooses a random $t_0 \in Z_N$ and random

elements $X'_3, Y'_3 \in G_{p_3}$ and responds with:

$$(w_0^s g_2^{\delta_2} v_0^{t_0} X'_3 \sigma_2, g^{t_0} g_2^{t_0} X'_3, (u_0^{T^*} h_0)^{t_0} g_2^{t_0(a' I_j^* + b')} Y'_3) \tag{60}$$

Lemma 23 *Under Assumptions 3, no PPT attacker can distinguish between* $\grave{O}^*_{k-1}$ *and* $\grave{O}'_k$ *with non-negligible advantage. So no PPT attacker can distinguish between* $L_{k-1,1}$ *and* $L_{k,2}$ *with non-negligible advantage.*

Lemma 24 *Under Assumptions 4, no PPT attacker can distinguish between* $\grave{O}'_k$ *and* $\grave{O}''_k$ *with non-negligible advantage. So no PPT attacker can distinguish between* $L_{k,2}$ *and* $L_{k,3}$ *with non-negligible advantage.*

Lemma 25 *Under Assumptions 3, no PPT attacker can distinguish between* $\grave{O}''_k$ *and* $\grave{O}^*_k$ *with non-negligible advantage. So no PPT attacker can distinguish between* $L_{k,3}$ *and* $L_{k,1}$ *with non-negligible advantage.*

From the Lemma 23, 24, 25, we obtain the following equation

$$\begin{aligned} Adv_{\mathcal{A}}^{G_{E-D}} - Adv_{\mathcal{A}}^{G_{ESF}} &\leq \sum_{k=0}^{q_c-1} (|Adv_{\mathcal{A}}^{L_{k-1,1}} - Adv_{\mathcal{A}}^{L_{k,2}}| + |Adv_{\mathcal{A}}^{L_{k,2}} - Adv_{\mathcal{A}}^{L_{k,3}}| \\ &\quad + |Adv_{\mathcal{A}}^{L_{k,3}} - Adv_{\mathcal{A}}^{L_{k,1}}|) \\ &\leq O(q_c)(2Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4}) \end{aligned} \tag{61}$$

(3) Indistinguishability of G_{ESF} and $G_{SF'}$

For the security proof of the indistinguishability of G_{ESF} and $G_{SF'}$, we define a sequence of games G_{ESF-1}, G_{ESF-2}, G_{ESF-3} to change the type of decryption keys from ESF-2 to ESF-4 and the type of ciphertexts from ESF-1 to ESF-5 and G_{ESF-4}, G_{ESF-5}, G_{ESF-6} to change the type of decryption keys to semi-functional and the type of ciphertexts back to semi-functional. In Table 8, we give the types of key in the queries and the challenge cipertext in every game, and the decryption situation according to the types of keys and ciphertexts.

G_{ESF-1}: The decryption keys are changed to ESF-3. The challenge ciphertext is still ESF-1. The secret keys and update keys are still semi-functional.

G_{ESF-2}: The challenge ciphertext is changed to ESF-5. The secret keys and the update keys are still semi-functional. The decryption keys are still ESF-3.

G_{ESF-3}: The decryption keys are changed to ESF-4. The challenge ciphertext is ESF-5. The secret keys and update keys are still semi-functional.

G_{ESF-4}: The challenge ciphertext is changed to ESF-1. The secret keys and the update keys are semi-functional. The decryption keys are still ESF-4.

G_{ESF-5}: The challenge ciphertext is changed to semi-functional. The secret keys and the update keys are semi-functional. The decryption keys are still ESF-4.

Table 8. Defination of games between G_{ESF} and $G_{SF'}$.

Games	Oracles	Keys in Queries			Challenge Ciphertext			
		SK	UK	DK	*Normal*	*SF*	*ESF* – 1	*ESF* – 5
G_{ESF}	O_6	*SF*	*SF*	*ESF* – 2			○	
G_{ESF-1}	$O_{6.1}$	*SF*	*SF*	*ESF* – 3			○	
G_{ESF-2}	$O_{6.2}$	*SF*	*SF*	*ESF* – 3				○
G_{ESF-3}	$O_{6.3}$	*SF*	*SF*	*ESF* – 4				○
G_{ESF-4}	$\ddot{O}^*_{q_c}$	*SF*	*SF*	*ESF* – 4			○	
G_{ESF-5}	$\ddot{O}^*_0$	*SF*	*SF*	*ESF* – 4		○		
$G_{SF'}$	O_7	*SF*	*SF*	*SF*		○		

https://doi.org/10.1371/journal.pone.0195204.t008

We firstly prove the indistinguishabilities between G_{ESF} to G_{ESF-1}, G_{ESF-1} to G_{ESF-5}. And then we prove the indistinguishability of G_{ESF-5} and $G_{SF'}$.

Indistinguishability of G_{ESF} and G_{ESF-1}. For the security proof of the indistinguishability of G_{ESF} and G_{ESF-1}, we define games $G_{F,1}, \cdots, G_{F,h_d}, \ldots, G_{F,q_d}$ where $G_{F,0} = G_{ESF}$ and $G_{F,q_d} = G_{ESF-1}$, and q_d is the number of decryption key queries of an adversary. In the game $G_{F,h}$ for $1 \leq h \leq q_d$, the challenge ciphertext is ESF-1, all (H)IBE private keys are semi-functional, the i^{st} queried decryption key where $i \leq h$ are of ESF-3, the remaining decryption keys with $i > h$ are ESF-2.

Oracle $O_{6.1}$ This oracle initializes in the same way as O_6 and provides the attacker with initial group elements from the same distribution. Upon receiving a challenge decryption-key-type query for I, $T \in Z_n$, it chooses $r, y', r', y'' \in Z_n$ randomly, and also chooses $X_2, X_2', Y_2, Y_2' \in G_{p_2}$ and $X_3, X_3', Y_3, Y_3' \in G_{p_3}$ randomly. It returns the group elements

$$\begin{aligned}&(w^{y'} g_3^{y'\psi_1}, g^{y'} g_3^{y'}, v^{y'}(u^I h)^r X_2 X_3, g^r Y_2 Y_3,\\ &w_0^{y''} g_3^{y''\psi_2}, g^{y''} g_3^{y''}, v_0^{y''}(u_0^T h_0)^{r'} X_2' X_3', g^{r'} Y_2' Y_3')\end{aligned} \tag{62}$$

to the attacker. It responds to a ciphertext-type query or a challenge (H)IBE-key-type query in the same way as O_6.

Lemma 26 *Under Assumptions 3, no PPT attacker can distinguish between O_6 and $O_{6.1}$ with non-negligible advantage. So no PPT attacker can distinguish between G_{F,h_d-1} and G_{F,h_d} with non-negligible advantage.*

Let $Adv_{\mathcal{A}}^{G_{F',h}}$ be the advantage of $\mathcal{A}$ in a game $G_{F',h}$. From the Lemma 27, we obtain the following equation

$$Adv_{\mathcal{A}}^{G_{ESF}} - Adv_{\mathcal{A}}^{G_{ESF-1}} \leq \sum_{h=1}^{q_d} (Adv_{\mathcal{A}}^{G_{F',h-1}} - Adv_{\mathcal{A}}^{G_{F',h}}) \leq O(q_d)(Adv_{\mathcal{B}}^{A3}) \tag{63}$$

Indistinguishability of G_{ESF-1} and G_{ESF-2}. For the security proof of the indistinguishability of G_{ESF-1} and G_{ESF-2}, we define the oracle below.

Oracle $O_{6.2}$ This oracle initializes a bit differently from the other oracles. It fixes random elements $g, u, h, v, w, u_0, h_0, v_0, w_0 \in G_{p_1}, g_2 \in G_{p_2}, g_3 \in G_{p_3}$. It chooses random exponents $s, \gamma, \delta_1, \delta_2, y, y_0, \psi, \sigma_1, \sigma_2, a', b', t_3, t_3', t_3'' \in Z_N$. It initially provides the attacker with the group elements:

$$\begin{aligned}&(g, u, h, v, w, g^s(g_2 g_3)^{\gamma}, w^y(g_2 g_3)^{y\psi}, g^y(g_2 g_3)^y, v^y(g_2 g_3)^{y\sigma_1},\\ &u_0, h_0, v_0, w_0, w_0^{y_0}(g_2 g_3)^{y_0\psi}, g^{y_0}(g_2 g_3)^{y_0}, v_0^{y_0}(g_2 g_3)^{y_0\sigma_2})\end{aligned} \tag{64}$$

What differs from the previous oracles here is the added g_3^{γ} and term: notice that this is uniformly random in G_{p_3}, since γ is random modulo p_3 (and uncorrelated from its value modulo p_2). This oracle answers the challenge-key type query in the same way as $O_{6.1}$. To answer a ciphertext-type query for I, it chooses random values $t \in Z_N$ and responds with:

$$(w^s g_2^{\delta_1} v^t g_2^{\sigma_1 t} g_3^{t_3}, g^t g_2^t g_3^{t'}, (u^I h)^t g_2^{t(a'I+b')} g_3^{t''}) \tag{65}$$

To answer a ciphertext-type query for T, it chooses random values $t \in Z_N$ and responds with:

$$(w_0^s g_2^{\delta_2} v_0^t g_2^{\sigma_2 t} g_3^{t_3}, g^t g_2^t g_3^{t'}, (u_0^T h_0)^t g_2^{t(a'T+b')} g_3^{t''}) \tag{66}$$

It is crucial to note that these G_{p_3} terms arethe same for each ciphertext-type query response.

Lemma 27 *Under Assumptions 4, no PPT attacker can distinguish between $O_{6.1}$ and $O_{6.2}$ with non-negligible advantage. So no PPT attacker can distinguish between G_{ESF-1} and G_{ESF-2} with non-negligible advantage.*

From the Lemma 27, we obtain the following equation

$$Adv_{\mathcal{A}}^{G_{ESF-1}} - Adv_{\mathcal{A}}^{G_{ESF-2}} \leq (Adv_{\mathcal{B}}^{A4}) \tag{67}$$

Indistinguishability of G_{ESF-2} and G_{ESF-3}: For the security proof of the indistinguishability of G_{ESF-2} and G_{ESF-3}, we define a sequence of games additional hybrid games $G_{F-2,1}, \ldots, G_{F-2,h}, \ldots, G_{F-2,q_d}$, where $G_{ESF-2} = G_{F-2,0}$ and q_d is the number of decryption key queries of an adversary. In the game $G_{F-2,h}$ for $1 \leq h \leq q_d$, the challenge ciphertext is ESF-5, all (H)IBE private keys are semi-functional, the i^{st} queried decryption key where $i \leq h$ are of ESF-4, the remaining decryption keys with $i > h$ are ESF-3.

Oracle $O_{6.3}$ This oracle initializes in the same way with $O_{6.2}$ and provides the attacker the same initial elements as $O_{6.2}$. This oracle answers the ciphertext-type query and (H)IBE key-type query in the same way as $O_{6.2}$. To answer a challenge decryption key type query for I, I, it chooses random values $y, r, y', r' \in Z_N, X_2, Y_2, X_2', Y_2' \in G_{p_2}$ randomly, and $X_3, Y_3, X_3', Y_3' \in G_{p_3}$ and responds with:

$$\begin{aligned} &(w^y(g_2g_3)^{y\psi_1}, g^y(g_2g_3)^y, v^y(u^Ih)^r X_2X_3, g^r Y_2Y_3, \\ &w_0^{y'}(g_2g_3)^{y'\psi_2}, g^{y'}(g_2g_3)^{y'}, v_0^{y'}(u_0^Th_0)^{r'} X_2'X_3', g^{r'} Y_2'Y_3') \end{aligned} \tag{68}$$

Lemma 28 *Under Assumptions 4, no PPT attacker can distinguish between $O_{6.2}$ and $O_{6.3}$ with non-negligible advantage. So no PPT attacker can distinguish between $G_{F-2,h}$ and $G_{F-2,h+1}$ with non-negligible advantage.*

Let $Adv_{\mathcal{A}}^{G_{F-2,h}}$ be the advantage of $\mathcal{A}$ in a game $G_{F-2,h}$. From the Lemma 28, we obtain the following equation

$$Adv_{\mathcal{A}}^{G_{ESF-2}} - Adv_{\mathcal{A}}^{G_{ESF-3}} \leq \sum_{h=1}^{q_d} (Adv_{\mathcal{A}}^{G_{F-2,h-1}} - Adv_{\mathcal{A}}^{G_{F-2,h}}) \leq O(q_d)(Adv_{\mathcal{B}}^{A4}) \tag{69}$$

Indistinguishability of G_{ESF-3} and G_{ESF-4}. For the security proof of the indistinguishability of G_{ESF-3} and G_{ESF-4}, we define the oracle below.

Oracle $\ddot{O}_i^*$ This oracle initializes in the same way as $\grave{O}_i^*$, and provides the attacker with initial group elements from the same distribution. It also responds to a ciphertext-type query as same as $\grave{O}_i^*$. It responds to the decryption-key-type and (H)IBE-key-type queries in the same way as $O_{6.3}$.

Lemma 29 *Under Assumptions 4, no PPT attacker can distinguish between $O_{6.3}$ and $\ddot{O}_{q_c}^*$ with non-negligible advantage. So no PPT attacker can distinguish between G_{ESF-3} and G_{ESF-4} with non-negligible advantage.*

From the Lemma 29, we obtain the following equation

$$Adv_{\mathcal{A}}^{G_{ESF-3}} - Adv_{\mathcal{A}}^{G_{ESF-4}} \leq (Adv_{\mathcal{B}}^{A4}) \tag{70}$$

Indistinguishability of G_{ESF-4} and G_{ESF-5}. We now prove the indistinguishability of G_{ESF-4} and G_{ESF-5} in a hybrid argument using polynomially many steps. We let q_c denote the number of ciphertext-type queries made by a PPT attacker $\mathcal{A}$. Firstly we define hybrid games $L'_{-1,1}, L'_{0,2}, L'_{0,3}, L'_{0,1}, L'_{1,2}, L'_{1,3}, L'_{1,1} \ldots, L'_{k,2}, L'_{k,3}, L'_{k,1}, \ldots, L'_{q_c-1,2}, L'_{q_c-1,3}, L'_{q_c-1,1}$, where $L'_{-1,1} = G_{ESF-4}$ and $L'_{q_c-1,1} = G_{ESF-5}$. The games are formally defined as follows:

Game $L'_{k,1}$ This game $L'_{k,1}$ for $0 \leq k \leq q_c$ is almost the same as G_{ESF-4} except the generation of the challenge ciphertext. The challenge ciphertext of $(T^*, I_1^*, \ldots, I_{q_c-1}^*)$ is generated as EST-2^k-CT outputed by **EncryptESF-2**^{k}.

Game $L'_{k,2}$ This game $L'_{k,2}$ for $0 \leq k \leq q_c - 1$ is almost the same as G_{ESF-4} except the generation of the challenge ciphertext. The challenge ciphertext of $(T^*, I_1^*, \ldots, I_{q_c-1}^*)$ is generated as EST-3^k-CT outputed by **EncryptESF-3**^{k}.

Game $L'_{k,3}$ This game $L'_{k,3}$ for $0 \leq k \leq q_c - 1$ is almost the same as G_{ESF-4} except the generation of the challenge ciphertext. The challenge ciphertext of $(T^*, I_1^*, \ldots, I_{q_c-1}^*)$ is generated as EST-4^k-CT outputed by **EncryptESF-4**^{k}.

We will define additional oracles $\acute{O}'_i$ for each i from 0 to $q_c - 1$, and $\acute{O}''_i$ for each i from 0 to $q_c - 1$.

Oracle $\acute{O}'_i$ This oracle acts the same as $\acute{O}^*_i$ except that its response to the ciphertext-type query is as same as $\grave{O}'_i$.

Oracle $\acute{O}''_i$ This oracle acts the same as $\acute{O}^*_i$ except that its response to the ciphertext-type query is as same as $\grave{O}''_i$.

Lemma 30 *Under Assumptions 3, no PPT attacker can distinguish between* $\acute{O}^*_k$ *and* $\acute{O}''_k$ *with non-negligible advantage. So no PPT attacker can distinguish between* $L'_{k,1}$ *and* $L'_{k,3}$ *with non-negligible advantage.*

Lemma 31 *Under Assumptions 4, no PPT attacker can distinguish between* $\acute{O}''_k$ *and* $\acute{O}'_k$ *with non-negligible advantage. So no PPT attacker can distinguish between* $L'_{k,3}$ *and* $L'_{k,2}$ *with non-negligible advantage.*

Lemma 32 *Under Assumptions 3, no PPT attacker can distinguish between* $\acute{O}'_k$ *and* $\acute{O}^*_{k-1}$ *with non-negligible advantage. So no PPT attacker can distinguish between* $L'_{k,2}$ *and* $L'_{k-1,1}$ *with non-negligible advantage.*

Let $Adv_{\mathcal{A}}^{L'_{k,1}}$, $Adv_{\mathcal{A}}^{L'_{k,2}}$ and $Adv_{\mathcal{A}}^{L'_{k,3}}$ be the advantage of $\mathcal{A}$ in the games $L'_{k,1}$, $L'_{k,2}$ and $L'_{k,3}$. From the Lemma 30, 31, 32, we obtain the following equation

$$\begin{aligned} Adv_{\mathcal{A}}^{G_{ESF-4}} - Adv_{\mathcal{A}}^{G_{ESF-5}} &\leq \sum_{k=0}^{q_c-1} (|Adv_{\mathcal{A}}^{L'_{k-1,1}} - Adv_{\mathcal{A}}^{L'_{k,2}}| + |Adv_{\mathcal{A}}^{L'_{k,2}} - Adv_{\mathcal{A}}^{L'_{k,3}}| \\ &\quad + |Adv_{\mathcal{A}}^{L'_{k,3}} - Adv_{\mathcal{A}}^{L'_{L,1}}|) \\ &\leq q_c(2Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4}) \end{aligned} \tag{71}$$

Indistinguishability of G_{ESF-5} **and** $G_{SF'}$. For the security proof of the indistinguishability of G_{ESF-5} and $G_{SF'}$, we define hybrid games $J'_{1,1}, J'_{1,2}, \ldots, J'_{h,1}, J'_{h,2}, \ldots, J'_{q_d,1}, J'_{q_d,2}$ where $J'_{0,2} = G_{ESF-5}$ and $J'_{q_d,2} = G_{SF'}$, and q_d is the number of decryption key queries of an adversary. The oracle and games are formally defined as follows:

Oracle $O_{13/2}$ This oracle initializes in the same way as $\acute{O}^*_0$ and provides the attacker with initial group elements from the same distribution. Upon receiving a challenge decryption-key-type query for I, $T \in Z_n$, it chooses $r, y', r', y'' \in Z_n$ randomly, and also chooses $X_2, Y_2, X'_2, Y'_2 \in G_{p_2}$ and $X_3, Y_3, X'_3, Y'_3 \in G_{p_3}$ randomly. It returns the group elements

$$\begin{aligned} &(w^{y'}(g_2g_3)^{y'\psi_1}, g^{y'}(g_2g_3)^{y'}, v^{y'}(u^I h)^r X_2X_3, g^r Y_3, \\ &w_0^{y''}(g_2g_3)^{y''\psi_2}, g^{y''}(g_2g_3)^{y''}, v_0^{y''}(u_0^T h_0)^{r'} X'_2X'_3, g^{r'} Y'_3) \end{aligned} \tag{72}$$

to the attacker. It responds to a ciphertext-type query or a challenge (H)IBE-key-type query in the same way as $\ddot{O}_0^*$.

Game $J'_{h,1}$ This game $J'_{h,1}$ for $1 \leq h \leq q_d$ is almost the same as G_{ESF-5} except the generation of decryption keys. The i^{st} queried decryption key is generated as follows:

1. $i < h$: It generates a semi-functional $DK_{ID|_k,T}$.
2. $i = h$: It generates a ESF-5 $DK_{ID|_k,T}$.
3. $i > h$: It generates a ESF-4 $DK_{ID|_k,T}$.

Game $J'_{h,2}$ This game $J'_{h,1}$ for $1 \leq h \leq q_d$ is almost the same as $J'_{h,1}$ except the generation of the h^{st} queried decryption key is generated as a semi-functional $DK_{ID|_k,T}$.

Lemma 33 *Under Assumptions 4, no PPT attacker can distinguish between* $\ddot{O}_0^*$ *and* $O_{13/2}$ *with non-negligible advantage. So no PPT attacker can distinguish between* $J'_{h-1,2}$ *and* $J'_{h,1}$ *with non-negligible advantage.*

Lemma 34 *Under Assumptions 3, no PPT attacker can distinguish between* $O_{13/2}$ *and* O_7 *with non-negligible advantage. So no PPT attacker can distinguish between* $J'_{h,1}$ *and* $J'_{h,2}$ *with non-negligible advantage.*

Let $Adv_{\mathcal{A}}^{G_{E',h}}$ be the advantage of $\mathcal{A}$ in a game $G_{E',h}$. From the Lemma 33, 34, we obtain the following equations

$$\begin{aligned} Adv_{\mathcal{A}}^{G_{ESF-5}} - Adv_{\mathcal{A}}^{G_{SF'}} &\leq \textstyle\sum_{h_c=1}^{q_d}(|Adv_{\mathcal{A}}^{J'_{h,1}} - Adv_{\mathcal{A}}^{J'_{h,2}}| + |Adv_{\mathcal{A}}^{J'_{h-1,2}} - Adv_{\mathcal{A}}^{J'_{h,1}}|) \\ &\leq O(q_d)(Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4}) \end{aligned}$$

So we obtain the following equation

$$\begin{aligned} Adv_{\mathcal{A}}^{G_{ESF}} - Adv_{\mathcal{A}}^{G_{SF'}} &\leq |(Adv_{\mathcal{A}}^{G_{ESF}} - Adv_{\mathcal{A}}^{G_{ESF-1}}| + \\ &\quad \textstyle\sum_{i=1}^{5} |(Adv_{\mathcal{A}}^{G_{ESF-i}} - Adv_{\mathcal{A}}^{G_{ESF-i+1}}| + |(Adv_{\mathcal{A}}^{G_{ESF-5}} - Adv_{\mathcal{A}}^{G_{SF'}}| \\ &\leq O(q_d)(Adv_{\mathcal{B}}^{A3}) + (Adv_{\mathcal{B}}^{A4}) + O(q_d)(Adv_{\mathcal{B}}^{A4}) + (Adv_{\mathcal{B}}^{A4}) + \\ &\quad q_c(2Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4}) + O(q_d)(Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4}) \\ &\leq (O(q_d) + O(l))(Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4}) \end{aligned} \tag{73}$$

According to the equations Eqs 56, 61, 73, we obtain the following equation

$$\begin{aligned} Adv_{\mathcal{A}}^{G_{SF''}} - Adv_{\mathcal{A}}^{G_{SF'}} &\leq |(Adv_{\mathcal{A}}^{G_{SF''}} - Adv_{\mathcal{A}}^{G_{E-D}}| \\ &\quad + |(Adv_{\mathcal{A}}^{G_{E-D}} - Adv_{\mathcal{A}}^{G_{ESF}}| + |(Adv_{\mathcal{A}}^{G_{ESF}} - Adv_{\mathcal{A}}^{G_{SF'}}| \\ &\quad \leq (O(q_d) + O(l))(Adv_{\mathcal{B}}^{A3} + Adv_{\mathcal{B}}^{A4}) \end{aligned} \tag{74}$$

4.6 Indistinguishability of G_C and $G_{C'}$ and Indistinguishability of G_{SF} and G_{SF}

Lemma 35 *Under Assumptions 3 and 4, for any PPT attacker* $\mathcal{A}$*, the difference in* $\mathcal{A}$*'s advantage between* G_θ *and* $G_{\theta'}$ *is negligible, where* $\theta \in \{C, SF\}$.

Proof We suppose there exists a PPT attacker $\mathcal{A}$ and a symbol of $\theta \in \{C, SF\}$ such that $\mathcal{A}$'s advantage changes non-negligibly between Game RHIBE$_\theta$ and Game RHIBE$_{\theta'}$. We will either

create a PPT algorithm $\mathcal{B}$ that breaks Assumption 3 with non-negligible advantage or a PPT algorithm $\mathcal{B}$ that breaks Assumption 4 with non-negligible advantage.

While playing Game RHIBE$_\theta$ under Type-1 adversary, $\mathcal{A}$ produces two values $I, I' \in Z_n$ which are unequal modulo n but are equal modulo p_3, with non-negligible probability. We let A denote $gcd(I - I', n)$, and we let B denote n/A. We then have that p_3 divides A, and $B \neq 1$.

While playing Game RHIBE$_\theta$ under Type-2 adversary, $\mathcal{A}$ produces two values $T, T' \in Z_n$ which are unequal modulo n but are equal modulo p_3, with non-negligible probability. We let A denote $gcd(T - T', n)$, and we let B denote n/A. We then have that p_3 divides A, and $B \neq 1$.

We consider two possible cases: 1) p_1 divides B and 2) $A = p_1p_3$, $B = p_2$. At least one of these cases must occur with non-negligible probability.

If case 1) occurs with non-negligible probability, we can create a $\mathcal{B}$ which breaks Assumption 3 with non-negligible advantage. $\mathcal{B}$ receives $g, g_2, X_1 X_3, T$. It can use these terms to simulate Game RHIBE$_\beta$ with $\mathcal{A}$ as follows. It picks values $a, b, c, d, a_0, b_0, c_0, d_0 \in Z_N$ uniformly at random and sets $u = g^a$, $h = g^b$, $v = g^c$, $w = g^d$, $u_0 = g^{a_0}$, $h_0 = g^{b_0}$, $v_0 = g^{c_0}$, $w_0 = g^{d_0}$, and gives $\mathcal{A}$ the following public parameters:

$$PP = (g, u, h, v, w, u_0, h_0, v_0, w_0, \Omega = e(g, g)^{\alpha})$$

We note that $\mathcal{B}$ knows the master secret key α, so it can easily make normal secret keys, normal update keys and normal decryption keys. Since $\mathcal{B}$ also knows g_2, it can easily make semi-functional ciphertexts. So $\mathcal{B}$ can play Game RHIBE$_C$ and Game RHIBE$_{C'}$ with $\mathcal{A}$.

To make the three kinds of semi-functional keys, $\mathcal{B}$ uses X_1, X_3 and g_2. More precisely, to make a semi-functional decryption key for T and $(I_1, \cdots, I_j)$, it also chooses random values $y_1, \cdots, y_{j-1}, y'_j, r_0, r_1, \cdots \cdots, r_j \in Z_n$. It forms the decryption key as:

$$D_{0,0} = g^{\alpha - \sum_{i=1}^{j} \lambda_i} (X_1X_3g_2)^{d_0 y'_{0,\theta}}, D_{0,1} = (X_1X_3g_2)^{y'_{0,\theta}}, D_{0,2} = g^{r_{0,\theta}},$$

$$D_{0,3} = (X_1X_3g_2)^{c_0 y'_{0,\theta}} (u_0^T h_0)^{r_{0,\theta}}$$

$$D_{i,0} = g^{\lambda_i} w^{y_{i,\theta}}, D_{i,1} = g^{y_{i,\theta}}, D_{i,2} = g^{r_{i,\theta}}, U_{i,3} = (u^{I_i} h)^{r_{i,\theta}} v^{y_{i,\theta}}, i \in \{1, \cdots, j-1\}$$

$$D_{j,0} = g^{\lambda_j} (X_1X_3g_2)^{dy'_{j,\theta}}, D_{j,1} = (X_1X_3g_2)^{y'_{j,\theta}}, D_{j,2} = g^{r_{j,\theta}},$$

$$D_{j,3} = (X_1X_3g_2)^{cy'_{j,\theta}} (u^{I_j} h)^{r_{j,\theta}}$$

To make the semi-functional update key of $(I_1, \cdots, I_{j-1})$ and T for each θ with its value γ_θ in KUNode($BT_{ID|_{j-1}}$, T, $RL_{ID|_{j-1}}$), $\mathcal{B}$ chooses random values $y_{1,\theta}, \cdots, y_{j-2,\theta}, y'_{0,\theta}, r_{0,\theta}, r_{1,\theta}, \cdots, r_{j-1,\theta} \in Z_n$. $\mathcal{B}$ forms the challenge update key for $\mathcal{A}$ as:

$$U_{0,0} = g^{\alpha - \gamma_\theta - \sum_{i=1}^{j-1} \lambda_i} (X_1X_3g_2)^{d_0 y'_{0,\theta}}, U_{0,1} = (X_1X_3g_2)^{y'_{0,\theta}}, U_{0,2} = g^{r_{0,\theta}},$$

$$U_{0,3} = (X_1X_3g_2)^{c_0 y'_{0,\theta}} (u_0^T h_0)^{r_{0,\theta}}$$

$$U_{i,0} = g^{\lambda_i} w^{y_{i,\theta}}, U_{i,1} = g^{y_{i,\theta}}, U_{i,2} = g^{r_{i,\theta}}, U_{i,3} = (u^{I_i} h)^{r_{i,\theta}} v^{y_{i,\theta}}, i \in \{1, \cdots, j-1\}$$

To make the semi-functional secret key of $(I_1, \cdots, I_j)$ and T for each θ with its value γ_θ in Path($ID|_j$), $\mathcal{B}$ chooses random values $y_{1,\theta}, \cdots, y_{j-1,\theta}, y'_{j,\theta}, r_{1,\theta}, \cdots, r_{j,\theta} \in Z_n$. $\mathcal{B}$ forms the

challenge secret key for $\mathcal{A}$ as:

$$K_{i,0} = g^{\lambda_i} w^{y_{i,\theta}}, K_{i,1} = g^{y_{i,\theta}}, K_{i,2} = g^{r_{i,\theta}}, K_{i,3} = (u^{I_i} h)^{r_{i,\theta}} v^{y_{i,\theta}}, i \in \{1, \cdots, j-1\}$$

$$K_{j,0} = g^{\gamma_\theta - \sum_{i=1}^{j-1} \lambda_i} (X_1 X_3 g_2)^{dy'_{j,\theta}}, K_{j,1} = (X_1 X_3 g_2)^{y'_{j,\theta}}, K_{j,2} = g^{r_{j,\theta}},$$

$$K_{j,3} = (X_1 X_3 g_2)^{cy'_{j,\theta}} (u^{I_j} h)^{r_{j,\theta}}$$

So $\mathcal{B}$ can play Game RHIBE$_{SF}$ and Game RHIBE$_{SF'}$ with $\mathcal{A}$. Now, if $\mathcal{A}$ fails to produce I, I' or T, T' such that $gcd(I - I', n) = A$ or $gcd(T - T', n) = A$ is divisible by p_3 and p_1 divides $B = n/A$, then $\mathcal{B}$ guesses randomly. However, with non-negligible probability, $\mathcal{A}$ will produce such an I, I' or T, T'. $\mathcal{B}$ can detect this by computing $A = gcd(I - I', n)$ or $A = gcd(T - T', n)$ and $B = n/A$, checking that g^B is the identity element (this will occur only if p_1 divides B since g has order p_1 in G) and checking that $(X_1 X_3)^B \neq 1$ (this confirms that p_3 does not divide B, hence it must divide A). When $\mathcal{B}$ detects this situation, it can test whether $T \in G_{p_1}$ or $T \in G_{p_1 p3}$ by testing if T^B is 1. If $T^B = 1$ holds, then $T \in G_{p_1}$. If $T^B \neq 1$, then $T \in G_{p_1 p3}$. Thus, $\mathcal{B}$ achieves non-negligible advantage in breaking Assumption 3.

If case 2) occurs with non-negligible probability, we can create a $\mathcal{B}$ which breaks Assumption 4 with non-negligible advantage. $\mathcal{B}$ receives g, g_3, $X_1 X_2$, $Y_2 Y_3$, T. It can use these terms to simulate Game RHIBE$_\theta$ with $\mathcal{A}$ as follows. It gives $\mathcal{A}$ the public parameters like in the case 1. We note that B knows the master secret key α, so it can easily make normal keys.

To make a semi-functional ciphertext for T and $(I_1^*, \cdots, I_l^*)$ and message M, $\mathcal{B}$ chooses random values $t_0, t_1, \cdots, t_l \in Z_n$ and forms the ciphertext as:

$$C = Me(X_1 X_2, g)^{\alpha}, C_0 = X_1 X_2,$$

$$C_{0,1} = (X_1 X_2)^{d_0} g^{c_0 t_0}, C_{0,2} = g^{a_0 T^* + b_0}, C_{0,3} = g^{t_0}$$

$$C_{i,1} = (X_1 X_2)^{d} g^{ct_i}, C_{i,2} = g^{aI_i^* + b}, C_{i,3} = g^{t_i}, i \in \{1, \cdots, l\}$$

We note that this will set $\sigma_1 = c_1$ modulo p_2 and $\sigma_2 = c_2$ modulo p_2. To make a semi-functional decryption key for $(I_1, \cdots, I_j)$, $\mathcal{B}$ chooses random values $y_0, y_1, \cdots, y_j, r_0, r_1, \cdots, r_j \in Z_n$. It forms the key as:

$$D_{0,0} = g^{\alpha - \sum_{i=1}^{j} \lambda_i} w_0^{y_0} (Y_2 Y_3)^{y_{0,\theta} \psi_2}, D_{0,1} = g^{y_0} (Y_2 Y_3)^{y_0}, D_{0,2} = g^{r_0},$$

$$D_{0,3} = v_0^{y_0} (Y_2 Y_3)^{y_0 \sigma_2} (u_0^T h_0)^{r_0}$$

$$D_{i,0} = g^{\lambda_i} w^{y_i}, D_{i,1} = g^{y_i}, D_{i,2} = g^{r_i}, D_{i,3} = (u^{I_i} h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

$$D_{j,0} = g^{\lambda_j} w^{y_{j,\theta}} (Y_2 Y_3)^{y_j \psi_1}, K_{j,1} = g^{y_j} (Y_2 Y_3)^{y_j}, D_{j,2} = g^{r_j},$$

$$D_{j,3} = v^{y_j} (Y_2 Y_3)^{y_j \sigma_1} (u^{I_j} h)^{r_j}$$

To make the semi-functional update key of $(I_1, \cdots, I_{j-1})$ and T for each θ with its value γ_θ in KUNode($BT_{ID|_{j-1}}$, T, $RL_{ID|_{j-1}}$), $\mathcal{B}$ chooses random values $y_{0,\theta}, y_{1,\theta}, \cdots, y_{j-1,\theta}$,

$r_{0,\theta}, r_{1,\theta}, \cdots, r_{j-1,\theta} \in Z_n$. $\mathcal{B}$ forms the challenge update key for $\mathcal{A}$ as:

$$U_{0,0} = g^{\alpha-\gamma_\theta-\sum_{i=1}^{j-1}\lambda_i} w_0^{y_{0,\theta}} (Y_2Y_3)^{y_{0,\theta}\psi_2}, U_{0,1} = g^{y_{0,\theta}} (Y_2Y_3)^{y_{0,\theta}}, U_{0,2} = g^{r_{0,\theta}},$$

$$U_{0,3} = v_0^{y_{0,\theta}} (Y_2Y_3)^{y_{0,\theta}\sigma_2} (u_0^T h_0)^{r_{0,\theta}}$$

$$U_{i,0} = g^{\lambda_i} w^{y_{i,\theta}}, U_{i,1} = g^{y_{i,\theta}}, U_{i,2} = g^{r_{i,\theta}}, U_{i,3} = (u^{I_i} h)^{r_{i,\theta}} v^{y_{i,\theta}}, i \in \{1, \cdots, j-1\}$$

To make the semi-functional secret key of $(I_1, \cdots, I_j)$ and T for each θ with its value γ_θ in Path$(ID|_j)$, $\mathcal{B}$ chooses random values $y_{1,\theta}, \cdots, y_{j,\theta}, r_{1,\theta}, \cdots, r_{j,\theta} \in Z_n$. $\mathcal{B}$ forms the challenge secret key for $\mathcal{A}$ as:

$$K_{i,0} = g^{\lambda_i} w^{y_{i,\theta}}, K_{i,1} = g^{y_{i,\theta}}, K_{i,2} = g^{r_{i,\theta}}, K_{i,3} = (u^{I_i} h)^{r_{i,\theta}} v^{y_{i,\theta}}, i \in \{1, \cdots, j-1\}$$

$$K_{j,0} = g^{\gamma_\theta-\sum_{i=1}^{j-1}\lambda_i} w^{y_{j,\theta}} (Y_2Y_3)^{y_{j,\theta}\psi_1}, K_{j,1} = g^{y_{j,\theta}} (Y_2Y_3)^{y_{j,\theta}}, K_{j,2} = g^{r_{j,\theta}},$$

$$K_{j,3} = v^{y_{j,\theta}} (Y_2Y_3)^{y_{j,\theta}\sigma_1} (u^{I_j} h)^{r_{j,\theta}}$$

We note that the semi-functional ciphertext and keys are well-distributed, and share the common value of $\sigma_1 = c_1$ modulo p_2 and $\sigma_2 = c_2$ modulo p_2 as required. We note that the G_{p_2} terms on the ciphertext are random because the value of d modulo p_2 and d_0 modulo p_2 does not appear elsewhere.

Now, if $\mathcal{A}$ fails to produce I, I' such that $gcd(I - I', n) = A$ or T, T' such that $gcd(T - T', n) = A$, where $A = p_1 p_3$ and $B = p_2$, then $\mathcal{B}$ guesses randomly. However, with non-negligible probability, $\mathcal{A}$ will produce such an I, I' or T, T'. $\mathcal{B}$ can detect this by computing A, B and testing that g^B and g_3^B are not the identity element (this confirms that $B = p_2$, since it demonstrates the p_1 and p_3 do not divide B). Now, $\mathcal{B}$ can learn whether T has a G_{p_2} component or not by testing if T^A is the identity element or not. If it is not, then T has a G_{p_2} component. Thus, $\mathcal{B}$ achieves non-negligible advantage in breaking Assumption 4.

4.7 Indistinguishability of G_{Real} and G_C

Lemma 36 *If the Assumption 1 holds, then no polynomial-time adversary can distinguish G_{Real} and G_C.*

Proof. We assume there is a PPT attacker $\mathcal{A}$ such that $\mathcal{A}$ achieves a non-negligible difference in advantage between Game G_{Real} and Game G_C. We will create a PPT algorithm $\mathcal{B}$ which breaks Assumption 1 with non-negligible advantage. $\mathcal{B}$ is given $g \in G_{p_1}$ and T. $\mathcal{B}$ chooses a, b, c, d, a_0, b_0, c_0, d_0, α randomly from $\mathbf{Z}_p$ and set $u = g^a, h = g^b, v = g^c, w = g^d, u_0 = g^{a_0}, h_0 = g^{b_0}$, $v_0 = g^{c_0}, w_0 = g^{d_0}$. It gives the public parameters

$$PP = (g, u, h, v, w, u_0, h_0, v_0, w_0, \Omega = e(g, g)^\alpha) \tag{75}$$

to $\mathcal{A}$. Since $\mathcal{B}$ knows the master secret key α, it can respond to $\mathcal{A}$'s key requests by calling the key generation update and derive algorithm and giving $\mathcal{A}$ the resulting keys.

At some point, $\mathcal{A}$ provides two messages M_0, M_1 and requests the challenge ciphertext for some identity vector, denoted by $(I_1^*, \ldots, I_l^*)$ at the time T^*. $\mathcal{B}$ forms the ciphertext as follows. It chooses $t_0, t_1, \ldots t_l$ randomly from $\mathbf{Z}_p$ and β randomly from {0, 1} and sets:

$$CT = (M_\beta e(g, T)^\alpha, T, T^{d_0} v_0^{t_0}, (u_0^{T^*} h_0)^{t_0}, g^{t_0}, \{T^d v^{t_i}, (u^{I_i^*} h)^{t_i}, g^{t_i}\}_{i=1}^{l}) \tag{76}$$

This implicitly sets g^s equal to the G_{p_1} part of T. If $T \in G_{p_1}$, then this is a well-distributed normal ciphertext, and $\mathcal{B}$ has properly simulated Game G_{Real}. If $T \in G_{p_1 p_2}$, then this is a well-

distributed semi-functional ciphertext (since the values of d modulo p_2 and d_0 modulo p_2 are uncorrelated from their values modulo p_1 by the Chinese Remainder Theorem). Hence, $\mathcal{B}$ has properly simulated Game G_C in this case. Thus, $\mathcal{B}$ can use the output of $\mathcal{A}$ to achieve a non-negligible advantage against Assumption 1.

4.8 Indistinguishability of G_{SF} and G_{Final}

Lemma 37 *If the Assumption 2 holds, then no polynomial-time adversary can distinguish G_{SF} and G_{Final}.*

Proof We suppose there exists a PPT attacker $\mathcal{A}$ who achieves a non-negligible advantage in Game RHIBE_{SF}. We will create a PPT algorithm $\mathcal{B}$ which has a non-negligible advantage against Assumption 2.

$\mathcal{B}$ receives $g, g_2, g_3, g^{\alpha} X_2, g^{s} Y_2, T$. It chooses $a, b, c, d, a_0, b_0, c_0, d_0$ randomly from $\mathbf{Z}_p$ and sets $u = g^a, h = g^b, v = g^c, w = g^d, u_0 = g^{a_0}, h_0 = g^{b_0}, v_0 = g^{c_0}, w_0 = g^{d_0}$. It gives the public parameters

$$PP = (g, u, h, v, w, u_0, h_0, v_0, w_0, \Omega = e(g, g^{\alpha} X_2)) \tag{77}$$

to $\mathcal{A}$. We note that $\mathcal{B}$ does not know the master secret key α. For a secret key query for $(I_1, \cdots, I_k)$, $\mathcal{B}$ will create a semi-functional secret key as follows. It chooses f_1 randomly and $r_{1,\theta}, \cdots, r_{k,\theta}, b_{1,\theta}, \cdots, b_{k,\theta} \in \mathbf{Z}_p$ randomly for each node $\theta \in Path(ID_k)$. The semi-functional secret key $SK_{ID|_k}$ is formed as $(\{\theta, PSK_\theta\}_{\theta \in path})$, in which $PSK_\theta = \left(\{\widetilde{K}_{i,0}, \widetilde{K}_{i,1}, \widetilde{K}_{i,2}, \widetilde{K}_{i,3}\}_{i=1}^{k}\right)$ and we have $(\widetilde{K}_{i,0}, \widetilde{K}_{i,1}, \widetilde{K}_{i,2}, \widetilde{K}_{i,3})$ as

$$\begin{cases} (g^{\lambda_i} w^{b_{i,\theta}}, g^{b_{i,\theta}}, g^{r_{i,\theta}}, (u^{I_i} h)^{r_{i,\theta}} v^{b_{i,\theta}}), & 0 \le i \le k-1 \\ (g^{\gamma_\theta - \prod_{i=1}^{k-1} \lambda_i} w^{b_{j,\theta}} (g_2 g_3)^{f_1 (d+1)}, g^{b_{i,\theta}} (g_2 g_3)^{f_1}, g^{r_{i,\theta}}, (u^{I_i} h)^{r_{i,\theta}} v^{b_{i,\theta}} (g_2 g_3)^{f_1 c}), & i = k \end{cases} \tag{78}$$

This is a well-distributed semi-functional secret key with $\psi_{1,\theta} = d + 1$, $\sigma_{1,\theta} = c (mod\ p_2 p_3)$ and $y_1 = f_1 (mod\ p_2 p_3)$. Notice that y_1 is freshly random modulo p_2 and p_3 for each key, while $\sigma_{2,\theta}$, $\psi_{2,\theta}$ are the same for all update keys.

For an update key query for $(I_1, \cdots, I_{j-1})$ and T, $\mathcal{B}$ generates a semi-functional update key as follows. It chooses $r_{1,\theta}, \ldots, r_{j,\theta}, b_{1,\theta}, \ldots, b_{j-1,\theta}, b'_{0,\theta} \in \mathbf{Z}_p$ randomly for each node $\theta \in KUNode$ $(BT_{ID_{j-1}})$ and f_2 randomly. And it will implicitly set $b_{0,\theta} = (b'_{0,\theta} + \alpha)\ mod\ p_1$. The semi-functional update key is formed as $UK_{ID|_{j-1},T} = (\{\theta, TUK_\theta\}_{\theta \in KUNode})$ and $TUK_\theta = (\{U_{i,0}, U_{i,1}, U_{i,2}, U_{i,3}\}_{i=0}^{j-1})$:

$$\begin{gathered} U_{0,0} = g^{-\gamma_\theta - \prod_{i=1}^{j-1} \lambda_i} (g^{\alpha} X_2)^{d_0+1} w_0^{b'_{0,\theta}} (g_2 g_3)^{f_2 (d_0+1)}, U_{0,1} = (g^{\alpha} X_2) g^{b'_{0,\theta}} (g_2 g_3)^{f_2}, \\ U_{0,2} = g^{r_{0,\theta}}, U_{0,3} = (g^{\alpha} X_2)^{c_0} (u_0^T h_0)^{r_{0,\theta}} v_0^{b'_{0,\theta}} (g_2 g_3)^{f_2 c_0} \\ U_{i,0} = g^{\lambda_i} w^{b_{i,\theta}}, U_{i,1} = g^{b_{i,\theta}}, U_{i,2} = g^{r_{i,\theta}}, U_{i,3} = (u^{I_i} h)^{r_{i,\theta}} v^{b_{i,\theta}}, i \in \{1, \cdots, j-1\} \end{gathered} \tag{79}$$

This is a well-distributed semi-functional update key with $\psi_{2,\theta} = d_0 + 1$, $\sigma_{2,\theta} = c_0 (mod\ p_2 p_3)$ and $y_2 = f_2 (mod\ p_3)$, $y_2 = (f_2 + log_{g_2} X_2) (mod\ p_2)$, then $X_2 (g_2 g_3)^{f_2} = X_2 (g_2)^{f_2} \cdot (g_3)^{f_2} = (g_2 g_3)^{y_2}$. Notice that y_2 is freshly random modulo p_2 and p_3 for each update key, while $\sigma_{2,\theta}$, $\psi_{2,\theta}$ are the same for all update keys.

In response to a decryption key query for $(I_1, \cdots, I_j)$ and T. $\mathcal{B}$ generates the semi-functional secret key and the semi-functional update key at first, and derives an semi-functional

decryption key which is formed as

$$D_{0,0} = g^{-\prod_{i=1}^{j}\lambda_i}(g^{\alpha}X_2)^{d_0+1}w_0^{b'_0}(g_2g_3)^{f_2(d_0+1)}, D_{0,1} = (g^{\alpha}X_2)g^{b'_0}(g_2g_3)^{f_2},$$

$$D_{0,2} = g^{r_0}, D_{0,3} = (g^{\alpha}X_2)^{c_0}(u_0^T h_0)^{r_0}v_0^{b'_0}(g_2g_3)^{f_2c_0}$$

$$D_{i,0} = g^{\lambda_i}w^{b_i}, D_{i,1} = g^{b_i}, D_{i,2} = g^{r_i}, D_{i,3} = (u^{I_i}h)^{r_i}v^{b_i}, i \in \{1, \cdots, j-1\} \quad (80)$$

$$D_{j,0} = g^{\lambda_j}w^{b_j}(g_2g_3)^{f_1(d+1)}, D_{j,1} = g^{b_j}(g_2g_3)^{f_1},$$

$$D_{j,2} = g^{r_j}, D_{j,3} = (u^{I_j}h)^{r_j}v^{b_j}(g_2g_3)^{f_1c}$$

This is a well-distributed semi-functional decryption key.

At some point, $\mathcal{A}$ provides $\mathcal{B}$ with two messages M_0, M_1, a challenge identity vector $(I_1^*, \cdots, I_l^*)$ and a challenge time T^*. $\mathcal{B}$ creates the challenge ciphertext as follows. It chooses $t_1, \cdots, t_l, \delta'_1, \delta'_2$ randomly from Z_n and β randomly from {0, 1} and sets:

$$C = M_\beta T, C_0 = g^s Y_2,$$

$$C_{0,1} = (g^s Y_2)^{d_0}v_0^{t_0}g_2^{\delta'_2}, C_{0,2} = (u_0^{T^*}h_0)^{t_0}, C_{0,3} = g^{t_0} \quad (81)$$

$$C_{i,1} = (g^s Y_2)^{d}v^{t_i}g_2^{\delta'_1}, C_{i,2} = (u^{I_j^*}h)^{t_i}, C_{i,3} = g^{t_i}, i \in \{1, \cdots, l\}$$

If $T = e(g, g)^{\alpha s}$, this is a well-distributed semi-functional encryption of M_β with $\gamma = log_{g_2}Y_2, \delta_1 = d \cdot log_{g_2}Y_2 + \delta'_1, \delta_2 = d_0 \cdot log_{g_2}Y_2 + \delta'_2$. Notice that δ'_1 and δ'_2 randomize these so that there is no correlation with d or d_0 modulo p_2. Hence this is uncorrelated from the exponents modulo p_2 of the semi-functional keys. In this case, $\mathcal{B}$ has properly simulated Game RHIBE_{SF}.

If T is a random element of G_T, then this is a semi-functional encryption of a random message, and hence the ciphertext contains no information about β. In this case, the advantage of $\mathcal{A}$ must be zero. Since we have assumed the advantage of $\mathcal{A}$ is non-negligible in Game RHIBE_{SF}, $\mathcal{B}$ can use the output of $\mathcal{A}$ to obtain a non-negligible advantage against Assumption 2.

This completes the proof of Theorem 1.

5 Conclusion

In this paper, we propose a RHIBE scheme by combining the unbounded LW-(H)IBE and the CS method in a modular way in composite bilinear groups. Moreover, our construction has the advantages of decryption key exposure resistance and short system public parameters. Since neither the naive dual system encryption for bounded RHIBEs nor the naive nested dual system encryption for unbounded HIBEs work in our unbounded RHIBE, we carefully redesign the hybrid games to show the information theoretic arguments successfully in the dual system encryption framework. Our RHIBE is the first unbounded RHIBE scheme that achieves the adaptive security.

A Defination of the ephemeral semi-functional ciphertexts and keys

In the defination of the first type of ephemeral semi-functional ciphertext, we add G_{p_2} term on every element of all ciphertext-element-groups. We define a sequence of type-2 ephemeral semi-functional ciphertexts with the index $0 \leq k \leq l$, every element of the first $k-1$ ciphertext-element-groups is in $G_{p_1p_2}$, and only the first elements of the rest of ciphertext-element-groups are added by G_{p_2} terms. In the defination of the third type of ephemeral semi-functional

ciphertext, every element of the first $i-1$ ciphertext-element-groups is in G_{p_1p2}; for the i^{st} ciphertext-element-group, the first element is in G_{p_1p2p3}, its rest elements are in G_{p_1p3}; and for the rest ciphertext-element-groups, we add G_{p_2} terms on the first elements of them. In the defination of the fourth type of ephemeral semi-functional ciphertext, every elements of the first $i-1$ ciphertext-element-groups are in G_{p_1p2}, every elements of the i^{st} ciphertext-element-group are in G_{p_1p2p3}, and for the rest ciphertext-element-groups, we add G_{p_2} terms on the first elements of them. In the defination of the fifth type of ephemeral semi-functional ciphertext, every element of all ciphertext-element-groups is in G_{p_1p2p3}.

EncryptESF-1 $(ID|_j, T, M, PP, \sigma_1, \sigma_2) \rightarrow \widetilde{CT}_{E-1}$ Let the normal ciphertext be $CT_{ID|_l} = (C, C_0, \{C_{i,1}, C_{i,2}, C_{i,3}\}_{i=0}^{l})$. It chooses $\gamma, \delta_1, \delta_2, a', b'$, and random $t_0, \ldots, t_j \in Z_n$ and forms the **ESF-1-CT** $\widetilde{CT}_{E-1}$ as

$$(C, C_0 \cdot g_2^{\gamma}, C_{0,1} \cdot g_2^{\delta_2+\sigma_2 t_0}, C_{0,2} g_2^{(a'T+b')t_0}, C_{0,3} g_2^{t_0}, \{C_{i,1} g_2^{\delta_1+\sigma_1 t_i}, C_{i,2} g_2^{(a'I_i+b')t_i}, C_{i,3} g_2^{t_i}\}_{i=1}^{j})$$

EncryptESF-2$^{\mathbf{k}}$ $(ID|_j, T, M, PP, \sigma_1, \sigma_2, k) \rightarrow \widetilde{CT}_{E-2^k}$ It chooses $\gamma, \delta_1, \delta_2, a', b'$, and random $t_0, \ldots, t_k \in Z_n$. It forms the first two elements and the first k element-groups of ESF-2$^{\mathrm{k}}$-CT as same as of ESF-1-CT, and the rest element-groups of ESF-2$^{\mathrm{k}}$-CT as same as of SF-CT.

EncryptESF-3$^{\mathbf{k}}$ $(ID|_j, T, M, PP, \sigma_1, \sigma_2, k) \rightarrow \widetilde{CT}_{E-3^k}$ It chooses $\gamma, \delta_1, \delta_2, a', b'$, and random $t_0, \ldots, t_k \in Z_n$, random $X_3, Y_3 \in G_{p_3}$. It forms the first two elements and the first $k-1$ element-groups of ESF-3$^{\mathrm{k}}$-CT as same as of ESF-1-CT, and the k^{st} element-group of ESF-3$^{\mathrm{k}}$-CT as

$$(C_{i,1} \cdot g_2^{\delta_1} X_3^{\sigma_1}, C_{i,2} Y_3, C_{i,3} X_3)$$

and the rest element-groups of ESF-3$^{\mathrm{k}}$-CT as same as of SF-CT.

EncryptESF-4$^{\mathbf{k}}$ $(ID|_j, T, M, PP, \sigma_1, \sigma_2, k) \rightarrow \widetilde{CT}_{E-3^k}$ It chooses $\gamma, \delta_1, \delta_2, a', b'$, and random $t_0, \ldots, t_k \in Z_n$, random $X_3, Y_3 \in G_{p_3}$. It forms the first two elements and the first $k-1$ element-groups of ESF-4$^{\mathrm{k}}$-CT as same as of ESF-1-CT, and the k^{st} element-group of ESF-4$^{\mathrm{k}}$-CT as

$$(C_{i,1} \cdot g_2^{\delta_1+\sigma_1 t_k} X_3^{\sigma_1}, C_{i,2} g_2^{(a'I_k+b')t_k} Y_3, C_{i,3} g_2^{t_k} X_3)$$

and the rest element-groups of ESF-4$^{\mathrm{k}}$-CT as same as of SF-CT.

EncryptESF-5 $(ID|_j, T, M, PP, \sigma_1, \sigma_2) \rightarrow \widetilde{CT}_{E-5}$ Let the normal ciphertext be $CT_{ID|_l} = (C, C_0, \{C_{i,1}, C_{i,2}, C_{i,3}\}_{i=0}^{l})$. It chooses $\gamma, \delta_1, \delta_2, a', b', g_3 \in G_{p_3}$, and random $t_0 \ldots t_j, t'_0 \ldots t'_j, t''_0 \ldots t''_j, t'''_0 \ldots t'''_j \in Z_n$. It forms the first two elements of $\widetilde{CT}_{E-5}$ as $C, C_0 \cdot g_2^{\gamma}$, and forms the element-groups of ESF-5-CT as

$$C_{i,1} \cdot g_2^{\delta_1+\sigma_1 t_k} g_3^{t'_i}, C_{i,2} g_2^{(a'I_k+b')t_i} g_3^{t''_i}, C_{i,3} g_2^{t_i} g_3^{t'''_i},$$

$$\{C_{i,1} \cdot g_2^{\delta_1+\sigma_1 t_k} g_3^{t'_i}, C_{i,2} g_2^{(a'I_k+b')t_i} g_3^{t''_i}, C_{i,3} g_2^{t_i} g_3^{t'''_i}\}_{i=1}^{j}$$

In the defination of the first type of ephemeral semi-functional secret key, we add G_{p_3} term on the last 2 elements of the last element-group. In the defination of the second type of ephemeral semi-functional secret key, we add G_{p_2p3} term on the last 2 elements of the last element-group. In the defination of the third type of ephemeral semi-functional secret key, we add G_{p_3} term on the first 2 elements of the last element-group and add G_{p_2p3} term on the last 2 elements of the last element-group. In the defination of the fourth type of ephemeral semi-functional secret key, every element of the last element-group is in G_{p_1p2p3}. In the defination of the fifth type of ephemeral semi-functional secret key, the first 2 elements and the last

element of the last element-group is in $G_{p_1 p2 p3}$, and the third element of the last element-group is in $G_{p_1 p3}$.

SKeyESF-1 $(ID|_j, ST_{ID|_{j-1}}, PP, \theta) \rightarrow \widetilde{PSK}_{E-1}$ Let the correlative component key to the node $\theta \in Path(ID_j)$ in the $BT_{ID|_{j-1}}$ be $PSK_{ID|_j,\theta} = (\{K_{i,0}, K_{i,1}, K_{i,2}, K_{i,3}\}_{i=1}^{j})$. It chooses random values X_3, $Y_3 \in G_{p_3}$ and forms the component ESF-1-SK $\widetilde{PSK}_{E-1}$ by changing the last element-group as

$$(K_{j,0}, K_{j,1}, K_{j,2}Y_3, K_{j,3}X_3)$$

SKeyESF-2 $(ID|_j, ST_{ID|_{j-1}}, PP, \theta) \rightarrow \widetilde{PSK}_{E-2}$ Let the correlative component key to the node $\theta \in Path(ID_j)$ in the $BT_{ID|_{j-1}}$ be $PSK_{ID|_j,\theta} = (\{K_{i,0}, K_{i,1}, K_{i,2}, K_{i,3}\}_{i=1}^{j})$. It chooses random values X_2, $Y_2 \in G_{p_2}$, X_3, $Y_3 \in G_{p_3}$ and forms the component ESF-2-SK $\widetilde{PSK}_{E-2}$ by changing the last element-group as

$$(K_{j,0}, K_{j,1}, K_{j,2}Y_2Y_3, K_{j,3}X_2X_3)$$

SKeyESF-3 $(ID|_j, ST_{ID|_{j-1}}, PP, \theta) \rightarrow \widetilde{PSK}_{E-3}$ It chooses chooses y', $r \in Z_p$ randomly, X_2, $Y_2 \in G_{p_2}$ randomly, and X_3, $Y_3 \in G_{p_3}$ randomly and forms the component seceret key ESF-3-SK $\widetilde{PSK}_{E-3}$ by constructing $\kappa(I_j, y', r)$ in the last element-group as

$$(w^{y'} g_3^{y'\psi}, g^{y'} g_3^{y'}, g^r Y_2 Y_3, v^{y'} (u^{I_j} h)^r X_2 X_3)$$

And the contruction of the other element-groups follows the construction of SK_{HIBE,S_θ} in RHIBE.GenKey.

SKeyESF-4 $(ID|_j, ST_{ID|_{j-1}}, PP, \theta) \rightarrow \widetilde{PSK}_{E-4}$ It chooses chooses y', $r \in Z_p$ randomly, X_2, $Y_2 \in G_{p_2}$ randomly, and X_3, $Y_3 \in G_{p_3}$ randomly and forms the component ESF-4-SK $\widetilde{PSK}_{E-4}$ by constructing $\kappa(I_j, y', r)$ in the last element-group as

$$(w^{y'} (g_2 g_3)^{y'\psi}, g^{y'} (g_2 g_3)^{y'}, g^r Y_2 Y_3, v^{y'} (u^{I_j} h)^r X_2 X_3)$$

And the contruction of the other element-groups follows the construction of SK_{HIBE,S_θ} in RHIBE.GenKey.

SKeyESF-5 $(ID|_j, ST_{ID|_{j-1}}, PP, \theta) \rightarrow \widetilde{PSK}_{E-5}$ It chooses chooses y', $r \in Z_p$ randomly, $X_2 \in G_{p_2}$ randomly, and X_3, $Y_3 \in G_{p_3}$ randomly and forms the component ESF-5-SK $\widetilde{PSK}_{E-5}$ by by constructing $\kappa(I_j, y', r)$ in the last element-group as

$$(w^{y'} (g_2 g_3)^{y'\psi}, g^{y'} (g_2 g_3)^{y'}, g^r Y_3, v^{y'} (u^{I_j} h)^r X_2 X_3)$$

And the contruction of the other element-groups follows the construction of SK_{HIBE,S_θ} in RHIBE.GenKey.

The constructions from the normal component update key to the (ephemeral) semi-functional component update keys are similar to that of secret keys, expect that we change the first element group of normal component update key to different types.

UKeyESF-1 $(T, ST_{ID|_{k-1}}, RL_{ID|_{k-1},T}, PP, \theta) \rightarrow \widetilde{TUK}_{E-1}$ Let the correlative component key to the node $\theta \in KUNode(RL_{ID|_{j-1},T})$ be $TUK_{ID|_{j-1},T,\theta} = (\{U_{i,0}, U_{i,1}, U_{i,2}, U_{i,3}\}_{i=0}^{j-1})$. It chooses random values $\widetilde{r}'_1, \widetilde{r}'_2 \in Z_n$ and forms the ephemeral semi-functional secret key $\widetilde{TUK}_{E-1}$ by changing the first element group as It chooses random values X_3, $Y_3 \in G_{p_3}$ and forms the component

ESF-1-SK $\widetilde{TUK}_{E-1}$ by changing the first element-group as

$$(U_{j,0}, U_{j,1}, U_{j,2}Y_3, U_{j,3}X_3)$$

UKeyESF-2 $(T, ST_{ID|_{k-1}}, RL_{ID|_{k-1},T}, PP, \theta) \rightarrow \widetilde{TUK}_{E-2}$ Let the correlative component key to the node $\theta \in KUNode(RL_{ID|_{j-1},T})$ be $TUK_{ID|_{j-1},T,\theta} = (\{U_{i,0}, U_{i,1}, U_{i,2}, U_{i,3}\}_{i=0}^{j-1})$. It chooses random values $\widetilde{r}'_1, \widetilde{r}'_2 \in Z_n$ and forms the ephemeral semi-functional secret key $\widetilde{TUK}_{E-2}$ by changing the first element group as

$$(U_0, U_{0,1}, U_{0,2}(g_2g_3)^{\widetilde{r}'_2}, U_{0,3}(g_2g_3)^{\widetilde{r}'_1})$$

UKeyESF-3 $(T, ST_{ID|_{k-1}}, RL_{ID|_{k-1},T}, PP, \theta) \rightarrow \widetilde{TUK}_{E-3}$ It chooses chooses $y', r \in Z_p$ randomly, $X_2, Y_2 \in G_{p_2}$ randomly, and $X_3, Y_3 \in G_{p_3}$ randomly and forms the component seceret key ESF-3-UK $\widetilde{TUK}_{E-3}$ by constructing $\kappa_T(T, y', r)$ of the first element-group as

$$(w_0^{y'}g_3^{y'\psi}, g^{y'}g_3^{y'}, g^rY_2Y_3, v_0^{y'}(u_0^Th_0)^rX_2X_3)$$

And the contruction of the other element-groups follows the construction of RSK_{HIBE} and SK_{IBE,S_θ} in RHIBE.UpdateKey.

UKeyESF-4 $(T, ST_{ID|_{k-1}}, RL_{ID|_{k-1},T}, PP, \theta) \rightarrow \widetilde{TUK}_{E-4}$ It chooses chooses $y', r \in Z_p$ randomly, $X_2, Y_2 \in G_{p_2}$ randomly, and $X_3, Y_3 \in G_{p_3}$ randomly and forms the component ESF-4-UK $\widetilde{TUK}_{E-4}$ by constructing $\kappa_T(T, y', r)$ in the first element-group as

$$(w_0^{y'}(g_2g_3)^{y'\psi}, g^{y'}(g_2g_3)^{y'}, g^rY_2Y_3, v_0^{y'}(u_0^Th_0)^rX_2X_3)$$

And the contruction of the other element-groups follows the construction of RSK_{HIBE} and SK_{IBE,S_θ} in RHIBE.UpdateKey.

UKeyESF-5 $(T, ST_{ID|_{k-1}}, RL_{ID|_{k-1},T}, PP, \theta) \rightarrow \widetilde{TUK}_{E-5}$ It chooses chooses $y', r \in Z_p$ randomly, $X_2 \in G_{p_2}$ randomly, and $X_3, Y_3 \in G_{p_3}$ randomly and forms the component ESF-5-UK $\widetilde{TUK}_{E-5}$ by by constructing $\kappa_T(T, y', r)$ in the first element-group as

$$(w_0^{y'}(g_2g_3)^{y'\psi}, g^{y'}(g_2g_3)^{y'}, g^rY_3, v_0^{y'}(u_0^Th_0)^rX_2X_3)$$

And the contruction of the other element-groups follows the construction of RSK_{HIBE} and SK_{IBE,S_θ} in RHIBE.UpdateKey.

DKeyESF-i $(ID|_j, T, MSK, RL_{ID|_{j-1},T}, PP) \rightarrow \widetilde{DK}_E$ The ephemeral semi-functional decryption key generation algorithm firstly retrieves $\theta^* \in KUNode(RL_{ID|_{j-1},T}) \bigcap Path(ID|_j)$, and gets $\widetilde{TUK}_E$ and $\widetilde{PSK}_E$ which are the correlative subkey to the node θ^* from $UKeyESF - i(T, ST_{ID|_{j-1}}, \theta^*)$ and $SKeyESF - i(ID|_j, \theta^*)$, and then forms the ephemeral semi-functional decryption key $\widetilde{DK}_E$ as same as **DeriveKeySF**.

B Proof of lemmas

Lemma 1 *Under Assumptions 3, no PPT attacker can distinguish between O_0 and $O_{1/2}$ with non-negligible advantage. So no PPT attacker can distinguish between $H_{h_c-1,2}$ and $H_{h_c,1}$ with non-negligible advantage.*

Proof We assume $\mathcal{B}$ interacts with one of O_0, $O_{1/2}$. $\mathcal{O}$ receives $g, g_2, X_1 X_3, Y_1 Y_3, T$. $\mathcal{O}$ will simulate either O_0 or $O_{1/2}$ with $\mathcal{B}$, depending on the value of T (which is either in G_{p_1} or $G_{p_1 p3}$). $\mathcal{O}$ picks values $a, b, c, d, a_0, b_0, c_0, d_0 \in Z_N$ uniformly at random and sets $u = g^a$, $h = g^b$, $v = g^c$,

$w = g^d$, $u_0 = g^{a_0}$, $h_0 = g^{b_0}$, $v_0 = g^{c_0}$, $w_0 = g^{d_0}$. $\mathcal{B}$ initially obtains the group elements

$$\begin{aligned} &g, u, h, v, w, g^s g_2^{\gamma}, (X_1X_3)^d g_2^{y_2 d}, (X_1X_3) g_2^{y_2}, (X_1X_3)^c g_2^{cy_2} \\ &u_0, h_0, v_0, w_0, (X_1X_3)^{zd_0} g_2^{y'_2 d_0}, (X_1X_3)^z g_2^{y'_2}, (X_1X_3)^{zc_0} g_2^{c_0 y'_2} \end{aligned} \tag{82}$$

from its oracle simulator who additionally chooses $s, \gamma, \delta_1, \delta_2, y_2, y'_2, z \in Z_N$ randomly.

We note that these are properly distributed, with y modulo p_1 implicitly set to the discrete logarithm of X_1 base g modulo p_1, equal to d modulo p_2 and p_3, y_0 modulo p_1 implicitly set to the discrete logarithm of Y_1 base g modulo p_1, equal to d_0 modulo p_2 and p_3, and σ equal to c modulo p_2 and p_3. Note that the values of c modulo p_1, p_2, p_3 are uncorrelated from each other by the Chinese Remainder Theorem, and $v = g_c$ only involves the value of c modulo p_1.

$\mathcal{B}$ chooses $\alpha \in Z_n$ randomly, and gives $\mathcal{A}$ the following public parameters:

$$PP = (g, u, h, v, w, u_0, h_0, v_0, w_0, \Omega = e(g, g)^{\alpha}) \tag{83}$$

We note that $\mathcal{B}$ knows the master secret key α. When $\mathcal{A}$ requests a normal update key or a normal decryption key, $\mathcal{B}$ can responds by using the usual key generation algorithm, since it knows α. And also $\mathcal{B}$ can respond the semi-functional keys according to the group elements in Eq 82 that have been offered by the oracle simulator.

When $\mathcal{A}$ requests the challenge ciphertext for messages M_0, M_1, identity vector $(I_1^*, \cdots, I_l^*)$ and T^*, $\mathcal{B}$ makes a ciphertext-type query to the oracle for each I_i^* and T^*. When $\mathcal{B}$ makes a ciphertext-type query for some identity I^*, $\mathcal{O}$ responds by choosing a random $t \in Z_N$ and returning $(w^s g_2^{\delta_1} v^t, g^t, (u^{I^*} h)^t)$ to $\mathcal{B}$ as same as Eq 6. When $\mathcal{B}$ makes a ciphertext-type query for some time T^*, $\mathcal{O}$ responds by choosing a random $t_0 \in Z_N$ and returning $(w_0^s g_2^{\delta_2} v_0^{t_0}, g^{t_0}, (u_0^{T^*} h_0)^{t_0})$ as same as Eq 7 to $\mathcal{B}$. Then $\mathcal{B}$ creats the semi-functional ciphertexts as

$$CT_{ID^*|_l, T^*} = (g^s g_2^{\gamma}, w_0^s g_2^{\delta_2} v_0^t, (u_0^{T^*} h_0)^t, g^t, \{w^s g_2^{\delta_1} v^{t_i}, (u^{I_i^*} h)^{t_i}, g^{t_i}\}_{i=1}^{l}),$$

When $\mathcal{B}$ creats the HIBE private key with the index pair (h, i_c) for some identity vector $(I_1, \cdots, I_j)$ in the index h node, the HIBE private key with an index pair (h, i_c) is generated as follows:

1. $i_c < h_c$: It randomly chooses $y_1, \cdots, y_j, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_{j-1}, r'_j, z, z' \in Z_n$ and generates a ESF-2-SK SK_{HIBE,θ_h}.

$$K_{i,0} = g^{\lambda_i} w^{y_i}, K_{i,1} = g^{y_i}, K_{i,2} = g^{r_i}, K_{i,3} = (u^{I_i} h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

$$K_{j,0} = g^{\gamma_\theta - \sum_{i=1}^{j-1} \lambda_i} w^{y_j}, K_{j,1} = g^{y_j}, K_{j,2} = v^{y_j} (X_1X_3)^{r'_j (aI_j + b)} g_2^z, K_{j,3} = (X_1X_3)^{r'_j} g_2^{z'}$$

   It implicitly sets g^{r_j} to be $X_1^{r'_j}$ and that is a properly distribution ESF-2-SK.

2. $i_c = h_c$: $\mathcal{B}$ chooses random values $y_1, \cdots, y_{j-1}, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_{j-1} \in Z_n$. $\mathcal{B}$ forms the challenge key as:

$$K_{i,0} = g^{\lambda_i} w^{y_i}, K_{i,1} = g^{y_i}, K_{i,2} = g^{r_i}, K_{i,3} = (u^{I_i} h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

$$K_{j,0} = g^{\gamma_\theta - \sum_{i=1}^{j-1} \lambda_i} T_0, K_{j,1} = T_1, K_{j,2} = T_3, K_{j,3} = T_2$$

   where (T_0, T_1, T_2, T_3) is the challenge HIBE key queried to $\mathcal{O}$ who chooses a random $y_0 \in Z_N$ and returns $(T_0, T_1, T_2, T_3) = (w^{y'}, g^{y'}, v^{y'} T^{aI+b}, T)$ to $\mathcal{B}$.

3. $i_c > h_c$: It simply generates a normal HIBE private key.

In the challenge HIBE key, it implicitly sets g^r to be the G_{p_1} part of T. If $T \in G_{p_1}$, then this matches the distribution of O_0 (since there are no G_{p_3} terms here), and so this will be a properly distributed normal key and $\mathcal{B}$ is playing Game $H_{h_c-1,2}$. If $T \in G_{p_1p3}$, then this matches the distribution of $O_{1/2}$ (note that a, b modulo p_2 are uniformly random and do not occur elsewhere- so there are random G_{p_3} terms attached to the last two group elements) and then $\mathcal{B}$ is playing Game $H_{h_c,1}$.

Hence, if a PPT attacker can distinguish between $H_{h_c-1,2}$ and $H_{h_c,1}$ with non-negligible advantage, $\mathcal{O}$ can distinguish between O_0 and $O_{1/2}$ with non-negligible advantage. It means $\mathcal{O}$ can gain a non-negligible advantage against Assumption 3.

Thus, under Assumptions 3, no PPT attacker can distinguish between O_0 and $O_{1/2}$ with non-negligible advantage and no PPT attacker can distinguish between $H_{h_c-1,2}$ and $H_{h_c,1}$ with non-negligible advantage.

Lemma 2 *Under Assumptions 4, no PPT attacker can distinguish between $O_{1/2}$ and O_1 with non-negligible advantage. So no PPT attacker can distinguish between $H_{h_c,1}$ and $H_{h_c,2}$ with non-negligible advantage.*

Proof We assume $\mathcal{B}$ interacts with one of $O_{1/2}$, and O_1. $\mathcal{O}$ receives $g, g_3, X_1 X_2, Y_2 Y_3, T$. $\mathcal{O}$ will simulate either $O_{1/2}$ or O_1 with $\mathcal{B}$, depending on the value of T (which is either in G or G_{p_1p3}). $\mathcal{O}$ picks values $a, b, c, d, a_0, b_0, c_0, d_0 \in Z_N$ uniformly at random and sets $u = g^a$, $h = g^b$, $v = g^c$, $w = g^d$, $u_0 = g^{a_0}$, $h_0 = g^{b_0}$, $v_0 = g^{c_0}$, $w_0 = g^{d_0}$. $\mathcal{B}$ initially obtains the group elements

$$\begin{gathered} g, u, h, v, w, X_1X_2, w^y(Y_2Y_3)^{y\psi_1}, g^y(Y_2Y_3)^y, v^y(Y_2Y_3)^{y\sigma_1}, \\ u_0, h_0, v_0, w_0, w_0^{y_0}(Y_2Y_3)^{y_0\psi_2}, g^{y_0}(Y_2Y_3)^{y_0}, v_0^{y_0}(Y_2Y_3)^{y_0\sigma_2} \end{gathered} \tag{84}$$

from its oracle simulator, where $y, \psi_1, \psi_2, \sigma_1, \sigma_2 \in Z_p$ are randomly chosen. It chooses $\alpha \in Z_n$ randomly, and gives $\mathcal{A}$ the public parameters in Eq 83.

When $\mathcal{A}$ requests the challenge ciphertext for messages M_0, M_1, identity vector $(I_1^*, \cdots, I_l^*)$ and T^*, $\mathcal{B}$ makes a ciphertext-type query to the oracle for each I_i^* and T^*. In response to each query for I_i^*, $\mathcal{O}$ responds $((X_1X_2)^d v^{t_i}, g^{t_i}, (u^{I_i^*}h)^{t_i})$ to $\mathcal{B}$ by choosing a random $t_i \in Z_N$. In response to the query for T^*, $\mathcal{O}$ responds $((X_1X_2)^{d_0} v_0^{t_0}, g^{t_0}, (u_0^{T^*}h_0)^{t_0})$ to $\mathcal{B}$ by choosing a random $t_0 \in Z_N$. Then $\mathcal{B}$ creats the semi-functional ciphertexts successfully.

When $\mathcal{B}$ creats the HIBE private key with the index pair (h, i_c) for some identity vector $(I_1, \cdots, I_j)$ in the index h node, the HIBE private key with an index pair (h, i_c) is generated as follows:

1. $i_c < h_c$: $\mathcal{B}$ chooses random values $y_1, \cdots, y_j, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_j, z, z' \in Z_n$ and generates a ESF-2 $PSK_{HIBE,h}$.

$$K_{i,0} = g^{\lambda_i} w^{y_i}, K_{i,1} = g^{y_i}, K_{i,2} = g^{r_i}, K_{i,3} = (u^{I_i}h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

$$K_{j,0} = g^{\gamma_\theta - \sum_{i=1}^{j-1} \lambda_i}(w^{y_j}, K_{j,1} = g^{y_j}, K_{j,2} = v^{y_j}(u^{I_j}h)^{r_j}(Y_2Y_3)^z, K_{j,3} = g^{r_j}(Y_2Y_3)^{z'}$$

2. $i_c = h_c$: $\mathcal{B}$ chooses random values $y_1, \cdots, y_{j-1}, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_{j-1} \in Z_n$. $\mathcal{B}$ forms the challenge key as:

$$K_{i,0} = g^{\lambda_i} w^{y_i}, K_{i,1} = g^{y_i}, K_{i,2} = g^{r_i}, K_{i,3} = (u^{I_i}h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

$$K_{j,0} = g^{\gamma_\theta - \sum_{i=1}^{j-1} \lambda_i} T_0, K_{j,1} = T_1, K_{j,2} = T_3, K_{j,3} = T_2$$

where (T_0, T_1, T_2, T_3) is the challenge HIBE key queried to $\mathcal{O}$ who chooses a random $y_0 \in Z_N$ and returns $(T_0, T_1, T_2, T_3) = (w^{y'}, g^{y'}, v^{y'} T^{aI+b}, T)$ to $\mathcal{B}$.

3. $i_c > h_c$: It simply generates a normal HIBE private key.

As in the previous lemma, this implicitly sets g_r to be the G_{p_1} part of T in the challenge HIBE key. We note that a, b modulo p_2, p_3 are uniformly random and do not appear elsewhere. Thus, when $T \in G_{p_1p3}$, these last two terms will have random elements of G_{p_3} attached (matching the distribution of $O_{1/2}$) and then $\mathcal{B}$ is playing Game $H_{h_c,1}$. And when $T \in G$, these last two terms will have random elements in both G_{p_3} and G_{p_2} attached (matching the distribution of O_1) and then $\mathcal{B}$ is playing Game $H_{h_c,2}$.

Hence, if a PPT attacker can distinguish between $H_{h_c,1}$ and $H_{h_c,2}$ with non-negligible advantage, $\mathcal{O}$ can distinguish between $O_{1/2}$ and O_1 with non-negligible advantage. It means $\mathcal{O}$ can gain a non-negligible advantage against Assumption 4.

Thus, Under Assumptions 4, no PPT attacker can distinguish between $O_{1/2}$ and O_1 with non-negligible advantage. Thus, no PPT attacker can distinguish between $H_{h_c,1}$ and $H_{h_c,2}$ with non-negligible advantage.

Lemma 3 *Under Assumptions 3, no PPT attacker can distinguish between O^*_{k-1} and O'_k with non-negligible advantage. So no PPT attacker can distinguish between $S_{k-1,1}$ and $S_{k,2}$ with non-negligible advantage.*

Proof We assume $\mathcal{B}$ interacts with one of O^*_{k-1}, O'_k. $\mathcal{O}$ receives g, g_2, X_1 X_3, Y_1 Y_3, T. $\mathcal{O}$ will simulate either O^*_{k-1} or O'_k with $\mathcal{B}$, depending on the value of T (which is either in G_{p_1} or G_{p_1p3}). $\mathcal{B}$ initially obtains the group elements in Eq 82

$$g, u, h, v, w, g^s g_2^{\gamma}, (X_1X_3)^d g_2^{y_2 d}, (X_1X_3) g_2^{y_2}, (X_1X_3)^c g_2^{cy_2}$$

$$u_0, h_0, v_0, w_0, (X_1X_3)^{zd_0} g_2^{y'_2 d_0}, (X_1X_3)^z g_2^{y'_2}, (X_1X_3)^{zc_0} g_2^{c_0 y'_2}$$

from its oracle simulator. It chooses $\alpha \in Z_n$ randomly, and gives $\mathcal{A}$ the public parameters in Eq 83. B can responds by using the normal update key generation and the normal decryption key derivation algorithm, since it knows α.

When $\mathcal{A}$ makes a secret key query for the identity $ID|_j = (I_1, \cdots, I_j)$, then $\mathcal{B}$ makes its challenge HIBE-key-type query for I_j, $\mathcal{O}$ responds as follows. It chooses y', r, r_1, $r_2 \in Z_N$ randomly and responds with:

$$(w^{y'}, g^{y'}, v^{y'} (X_1X_3)^{r(aI_j+b)} g_2^{r_1}, (X_1X_3)^r g_2^{r_2})$$

When $\mathcal{A}$ requests the challenge ciphertext for messages M_0, M_1, identity vector $(I^*_1, \cdots, I^*_l)$ and T^*, $\mathcal{B}$ makes a ciphertext-type query to the oracle for each I^*_i and T^*. In response to each query for I^*_i or T^*, $\mathcal{O}$ gets random $t_0, t_1, \ldots, t_l \in Z_p$, chooses $\beta \in \{0, 1\}$ and creats the ciphertext as

$$(C = M_\beta e(g^s g_2^{\gamma}, g)^{\alpha}, C_0 = g^s g_2^{\gamma}, \{C_{i,1}, C_{i,2}, C_{i,3}\}_{i=0}^{l})$$

where the ciphertext-element-group $(C_{i,1}, C_{i,2}, C_{i,3})$ is defined as follows:

1. $i < k$: If $i = 0$, $\mathcal{O}$ and responds with the ciphertext-element-group $(w_0^s g_2^{\delta_2} v_0^{t_0} g_2^{c_0 t_0}, (u_0^{T^*} h_0)^{t_0} g_2^{(a'T^*+b')t_0}, g^{t_0} g_2^{t_0})$, else the element group is $(w^s g_2^{\delta_1} v^{t_i} g_2^{ct_i}, (u^{I^*_i} h)^{t_i} g_2^{(a'I^*_i+b')t_i}, g^{t_i} g_2^{t_i})$;

2. $i = k$: The ciphertext-element-group is $(T_1, T_3, T_2) = (w^s g_2^{\delta_1} T^c, T^{(aI^*_i+b)}, T)$;

3. $i > k$: The ciphertext-element-group is $(w^s g_2^{\delta_1} v^{t_i}, (u^{I_i^*} h)^{t_i}, g^{t_i})$.

We must now argue that the challenge key-type query and the k^{th} ciphertext-type query responses are properly distributed. If $T \in G_{p_1}$, then the response to the k ciphertext type query is identically distributed to a response from O_1, and the values a, b modulo p_3 only appear in the response to the challenge key-type query, hence the G_{p_3} parts on the last two group elements here appear random in G_{p_3}. This will be a properly distributed EST-2^{k-1}-CT which means that the responses of $\mathcal{O}$ properly simulate the responses of O_{k-1}^* and $\mathcal{B}$ is playing Game $S_{k-1,1}$.

If $T \in G_{p_1 p3}$, then we must argue that $aI + b$ and $aI_k^* + b$ both appear to be uniformly random modulo p_3: this follows from pairwise independence of the function $aI + b$ modulo p_3, since we have restricted the **Type-1** adversary to choose I and I_k^* so that $I \neq I_k^*$ modulo p_3. This means that the G_{p_3} components on the last two group elements of the challenge key-type query response and on the k ciphertext-type query response are uniformly random in the attacker's view. In this case, $\mathcal{O}$ has produced a properly distributed EST-3^k-CT which means that $\mathcal{O}$ has properly simulated the responses of O_k' and $\mathcal{B}$ is playing Game $S_{k,2}$.

Particularly, we need overcome the paradox in the game hopping from Game $S_{q_c-1,1}$ to Game $S_{q_c 2}$ since the simulator can derive a decryption key and check whether the ciphertext is normal or semi-functional by being decrypted by the semi-functional derived decryption key from secret keys and update keys. For the game hopping from Game $S_{q_c-1,1}$ to Game $S_{q_c 2}$, no matter whether $T \in G_{p_1 p3}$ or $T \in G_{p_1}$, the cipertext- element-group (T_1, T_3, T_2) can be decrypted by the decryption key derived from the ESF-2-SK and normal update key. So the paradox is overcame successfully. (The other paradox need to overcome is in the game hopping from Game $L_{q_c-1,1}$ to Game $L_{q_c 2}$. In Lamma 23, the paradox can be overcame In the same way.)

Hence, if a PPT attacker can distinguish any pair between $S_{k-1,1}$ and $S_{k,2}$ with non-negligible advantage, $\mathcal{O}$ can distinguish the corresponding pair between O_{k-1}^* and O_k' with non-negligible advantage. It means $\mathcal{O}$ can use the output of $\mathcal{B}$ to achieve a non-negligible advantage against Assumption 3.

Thus, Under Assumptions 3, no PPT attacker can distinguish between O_{k-1}^* and O_k' with non-negligible advantage. Thus, no PPT attacker can distinguish between $S_{k-1,1}$ and $S_{k,2}$ with non-negligible advantage.

Lemma 4 *Under Assumptions 4, no PPT attacker can distinguish between O_k' and O_k'' with non-negligible advantage. So no PPT attacker can distinguish between $S_{k,2}$ and $S_{k,3}$ with non-negligible advantage.*

Proof We assume $\mathcal{B}$ interacts with one of O_k', O_k''. $\mathcal{O}$ receives g, g_3, X_1 X_2, Y_2 Y_3, T. $\mathcal{O}$ will simulate either O_k' or O_k'' with $\mathcal{B}$, depending on the value of T (which is either in G or $G_{p_1 p3}$). $\mathcal{O}$ picks values a, b, c, d, a_0, b_0, c_0, $d_0 \in Z_N$ uniformly at random and sets $u = g^a$, $h = g^b$, $v = g^c$, $w = g^d$, $u_0 = g^{a_0}$, $h_0 = g^{b_0}$, $v_0 = g^{c_0}$, $w_0 = g^{d_0}$. $\mathcal{B}$ initially obtains the group elements

$$\begin{aligned} &g, u, v, w, X_1 X_2, w^y (Y_2 Y_3)^{\psi}, g^y (Y_2 Y_3), v^y (Y_2 Y_3)^c, \\ &u_0, v_0, w_0, w_0^{y_0} (Y_2 Y_3)^{z\psi}, g^{y_0} (Y_2 Y_3)^z, v_0^{y_0} (Y_2 Y_3)^{zc_0} \end{aligned} \tag{85}$$

from its oracle simulator where z, y_0, y, $\psi \in Z_p$ are randomly chosen. These are properly distributed, with $g^s = X_1$ and $g_2^{\gamma} = X_2$. Note that this sets σ_1 equal to c modulo p_2 and p_3 and σ_2 equal to c_0 modulo p_2 and p_3. It chooses $\alpha \in Z_n$ randomly, and gives $\mathcal{A}$ the following public parameters in Eq 83. We note that $\mathcal{B}$ knows the master secret key α. When $\mathcal{A}$ requests a normal update key or a normal decryption key, $\mathcal{B}$ can responds by using the usual key generation algorithm, since it knows α.

When $\mathcal{A}$ makes a secret key query for the identity $ID|_j = (I_1, \cdots, I_j)$, then $\mathcal{B}$ makes its challenge HIBE-key-type query for I_j, $\mathcal{O}$ responds as follows. It chooses y', r, r_1, $r_2 \in Z_N$ randomly and responds with:

$$(w^{y'}, g^{y'}, v^{y'}(u^{I_j}h)^{r}(Y_2Y_3)^{r_1}, g^{r}(Y_2Y_3)^{r_2})$$

This has uniformly random terms in G_{p_2} and G_{p_3} on the last two elements, since r_1, r_2 are both uniformly random modulo p_2 and p_3.

When $\mathcal{A}$ requests the challenge ciphertext for messages M_0, M_1, identity vector $(I_1^*, \cdots, I_l^*)$ and T^*, $\mathcal{B}$ makes a ciphertext-type query to the oracle for each I_i^* and T^*. In response to each query for I_i^* or T^*, $\mathcal{O}$ gets random $t_0', t_1', \ldots, t_k', t_{k+1} \ldots, t_l \in Z_p$, chooses $\beta \in \{0, 1\}$ and creats the ciphertext as

$$(C = M_\beta e(X_1X_2, g)^{\alpha}, C_0 = X_1X_2, \{C_{i,1}, C_{i,2}, C_{i,3}\}_{i=0}^{l})$$

where the ciphertext-element-group $(C_{i,1}, C_{i,2}, C_{i,3})$ is defined as follows:

1. $i < k$: If $i = 0$, the ciphertext-element-group is

$$((X_1X_2)^{d_0}(X_1X_2)^{c_0t_0'}, (X_1X_2)^{t_0'(a_0T^*+b_0)}, (X_1X_2)^{t_0'})$$

else the ciphertext-element-group is

$$((X_1X_2)^{d}(X_1X_2)^{ct_i'}, (X_1X_2)^{t_i'(aI_i^*+b)}, (X_1X_2)^{t_i'})$$

This sets $X_2^{d} = g_2^{\delta_1}$ and $X_2^{d_0} = g_2^{\delta_2}$, which is uniformly random because the value of d and d_0 modulo p_2 will not appear elsewhere. It implicitly sets $g^{t_i} = X_1^{t_i'}$. This is identically distributed to a response from O_2, with a', b' equal to a, b modulo p_2, and $\sigma_1 = c$ modulo p_2, $\sigma_2 = c_0$ modulo p_2. We note that this is in the only context in which the values of a, b modulo p_2 appear, so this is equivalent to choosing a', b' independently at random.

2. $i = k$: If $k = 0$, the ciphertext-element-group is $(T_1, T_3, T_2) = ((X_1X_2)^{d_0}T^{c_0}, T^{(a_0T^*+b_0)}, T)$; If $k > 0$, the ciphertext-element-group is $(T_1, T_3, T_2) = ((X_1X_2)^{d}T^{c}, T^{(aI_i^*+b)}, T)$;

3. $i > k$: The ciphertext-element-group is $((X_1X_2)^{d}v^{t_i}, (u^{I_i^*}h)^{t_i}, g^{t_i})$.

If $T \in G_{p_1p3}$, then the response for ciphertext-type query i is identically distributed to a response from O_k'.

In this case, $\mathcal{O}$ has produced a properly distributed EST-3^{k}-CT and $\mathcal{B}$ is playing Game $S_{k,2}$.

If $T \in G$, then this response additionally has terms in G_{p_2} which are appropriately distributed with $c = \sigma_1$, $a = a'$, $b = b'$ modulo p_2 or $c_0 = \sigma_2.a_0 = a'$, $b_0 = b'$ modulo p_2. Thus, the response is identically distributed to a response from O_k''. In this case, $\mathcal{O}$ has produced a properly distributed EST-4^{k}-CT and $\mathcal{B}$ is playing Game $S_{k,3}$.

Hence, if a PPT attacker can distinguish any pair between $S_{k,2}$ and $S_{k,3}$ with non-negligible advantage, $\mathcal{O}$ can distinguish the corresponding pair between O_k' and O_k'' with non-negligible advantage. It means $\mathcal{O}$ can use the output of $\mathcal{B}$ to achieve a non-negligible advantage against Assumption 4.

Thus, Under Assumptions 4, no PPT attacker can distinguish between O_k' and O_k'' with non-negligible advantage. Thus, no PPT attacker can distinguish between $S_{k,2}$ and $S_{k,3}$ with non-negligible advantage.

Lemma 5 *Under Assumptions 3, no PPT attacker can distinguish between O_k'' and O_k^* with non-negligible advantage. So no PPT attacker can distinguish between $S_{k,3}$ and $S_{k,1}$ with non-negligible advantage.*

Proof We assume $\mathcal{B}$ interacts with one of O_k'', O_k^*. $\mathcal{O}$ receives $g, g_2, X_1 X_3, T$. $\mathcal{O}$ will simulate either O_k'' or O_k^* with $\mathcal{B}$, depending on the value of T (which is either in G_{p_1} or $G_{p_1 p3}$). $\mathcal{B}$ initially obtains the group elements in Eq 82

$$g, u, h, v, w, g^s g_2^{\gamma}, (X_1X_3)^d g_2^{y_2 d}, (X_1X_3) g_2^{y_2}, (X_1X_3)^c g_2^{cy_2}$$

$$u_0, h_0, v_0, w_0, (X_1X_3)^{zd_0} g_2^{y_2' d_0}, (X_1X_3)^z g_2^{y_2'}, (X_1X_3)^{zc_0} g_2^{c_0 y_2'}$$

from its oracle simulator. It chooses $\alpha \in Z_n$ randomly, and gives $\mathcal{A}$ the public parameters in Eq 83. B can responds by using the normal update key generation and the normal decryption key derivation algorithm, since it knows α.

When $\mathcal{A}$ makes a secret key query for the identity $ID|_j = (I_1, \cdots, I_j)$, then $\mathcal{B}$ makes its challenge HIBE-key-type query for I_j, $\mathcal{O}$ responds as follows. It chooses $y', r, r_1, r_2 \in Z_N$ randomly and responds with:

$$(w^{y'}, g^{y'}, v^{y'} (X_1X_3)^{r(aI_j+b)} g_2^{r_1}, (X_1X_3)^r g_2^{r_2})$$

We note that the G_{p_2} parts here are uniformly random.

When $\mathcal{A}$ requests the challenge ciphertext for messages M_0, M_1, identity vector $(I_1^*, \ldots, I_l^*)$ and T^*, $\mathcal{B}$ makes a ciphertext-type query to the oracle for each I_i^* and T^*. In response to each query for I_i^* or T^*, $\mathcal{O}$ gets random $t_0, t_1, \ldots, t_l \in Z_p$, chooses $\beta \in \{0, 1\}$ and creats the ciphertext as

$$(C = M_\beta e(g^s g_2^{\gamma}, g)^{\alpha}, C_0 = g^s g_2^{\gamma}, \{C_{i,1}, C_{i,2}, C_{i,3}\}_{i=0}^{l})$$

where the ciphertext-element-group $(C_{i,1}, C_{i,2}, C_{i,3})$ is defined as follows:

1. $i < k$: If $i = 0$, $\mathcal{O}$ responds with the ciphertext-element-group $(w_0^s g_2^{\delta_2} v_0^{t_0} g_2^{c_0 t_0}, (u_0^{T^*} h_0)^{t_0} g_2^{(a'T^*+b')t_0}, g^{t_0} g_2^{t_0})$, else the element group is $(w^s g_2^{\delta_1} v^{t_i} g_2^{ct_i}, (u^{I_i^*} h)^{t_i} g_2^{(a'I_i^*+b')t_i}, g^{t_i} g_2^{t_i})$. This is identically distributed to a response from O_2.
2. $i = k$: $\mathcal{O}$ choses $z \in Z_p$ randomly and responds with the ciphertext-element-group $(T_1, T_3, T_2) = (w^s g_2^{\delta_1} T^c \cdot g_2^{cz}, T^{(aI_i^*+b)} g_2^{z(a'I_i^*+b')}, T g_2^z)$ if $k > 0$. Else if $k = 0$, $\mathcal{O}$ responds with $(w_0^s g_2^{\delta_2} T^{c_0} \cdot g_2^{c_0 z}, T^{(a_0 T^*+b_0)} g_2^{z(a'T^*+b')}, T g_2^z)$. We note that the G_{p_2} parts here are properly distributed, since $\sigma_1 = c$ modulo p_2 and $\sigma_2 = c_0$ modulo p_2.
3. $i > k$: The ciphertext-element-group is $(w^s g_2^{\delta_1} v^{t_i}, (u^{I_i^*} h)^{t_i}, g^{t_i})$. This is identically distributed to a response from O_1.

When $T \in G_{p_1}$ the values of a, b modulo p_3 only appear in the response to the challenge key-type query, which means that the G_{p_3} terms on the last two group elements there are uniformly random. Also, the response to the k^{th} ciphertext-type query is distributed exactly like a response from O_2. In this case, $\mathcal{O}$ has properly simulated the responses of O_k^* and this will be a properly distributed EST-2^k-CT and so $\mathcal{B}$ is playing Game $S_{k,1}$.

When $T \in G_{p_1 p3}$, we must argue that the values $aI + b$ and $aI_k^* + b$ appear uniformly random modulo p_3: this follows by pairwise independence of $aI + b$ as a function of I modulo p_3, since we have restricted the **Type-1** adversary to choose I and I_k^* so that $I \neq I_k^*$ modulo p_3 and a, b modulo p_3 only appear in these two values. Hence, $\mathcal{O}$ has produced a properly distributed

EST-4^k-CT and $\mathcal{O}$ has properly simulated the response of O''_k in this case. So $\mathcal{B}$ is playing Game $S_{k,3}$. We have thus shown that $\mathcal{O}$ can use the output of $\mathcal{B}$ to achieve non-negligible advantage against Assumption 3.

Hence, if a PPT attacker can distinguish any pair between $S_{k,3}$ and $S_{k,1}$ with non-negligible advantage, $\mathcal{O}$ can distinguish the corresponding pair between O''_k and O^*_k with non-negligible advantage. It means $\mathcal{O}$ can use the output of $\mathcal{B}$ to achieve a non-negligible advantage against Assumption 3.

Thus, Under Assumptions 3, no PPT attacker can distinguish between O''_k and O^*_k with non-negligible advantage. Thus, no PPT attacker can distinguish between $S_{k,3}$ and $S_{k,1}$ with non-negligible advantage.

Lemma 6 *Under Assumptions 3, no PPT attacker can distinguish between O_2 and $O_{5/2}$ with non-negligible advantage. So no PPT attacker can distinguish between $E_{h_c-1,2}$ and $E_{h_c,1}$ with non-negligible advantage.*

Proof We assume $\mathcal{B}$ interacts with one of O_2 and $O_{5/2}$. $\mathcal{O}$ receives $g, g_2, X_1 X_3, Y_1 Y_3, T$. $\mathcal{O}$ will simulate either O_2 and $O_{5/2}$ with $\mathcal{B}$, depending on the value of T (which is either in G_{p_1} or $G_{p_1 p3}$). $\mathcal{O}$ picks values $a, b, c, d, a_0, b_0, c_0, d_0 \in Z_N$ uniformly at random and sets $u = g^a$, $h = g^b$, $v = g^c$, $w = g^d$, $u_0 = g^{a_0}$, $h_0 = g^{b_0}$, $v_0 = g^{c_0}$, $w_0 = g^{d_0}$. $\mathcal{B}$ initially obtains the group elements in Eq 82

$$g, u, h, v, w, g^s g_2^{\gamma}, (X_1X_3)^d g_2^{y_2 d}, (X_1X_3) g_2^{y_2}, (X_1X_3)^c g_2^{cy_2}$$

$$u_0, h_0, v_0, w_0, (X_1X_3)^{zd_0} g_2^{y'_2 d_0}, (X_1X_3)^z g_2^{y'_2}, (X_1X_3)^{zc_0} g_2^{c_0 y'_2}$$

from its oracle simulator who additionally chooses $s, \gamma, \delta_1, \delta_2, y_2, y'_2 \in Z_N$ randomly.

$\mathcal{B}$ chooses $\alpha \in Z_n$ randomly, and gives $\mathcal{A}$ the public parameters in Eq 83. We note that $\mathcal{B}$ knows the master secret key α. When $\mathcal{A}$ requests a normal decryption key, $\mathcal{B}$ can responds by using the usual key generation algorithm, since it knows α.

When $\mathcal{A}$ requests the challenge ciphertext for messages M_0, M_1, identity vector $(I^*_1, \ldots, I^*_l)$ and T^*, $\mathcal{B}$ makes a ciphertext-type query to the oracle for each I^*_i and T^*. When $\mathcal{B}$ makes a ciphertext-type query for some identity I^*, $\mathcal{O}$ responds by choosing a random $t \in Z_N$ and returning $(w^s g_2^{\delta_1} v^t g_2^{\sigma_1 t}, g^t g_2^t, (u^{I^*} h)^t g_2^{t(a'I^* + b')})$ to $\mathcal{B}$ as same as Eq 10. When $\mathcal{B}$ makes a ciphertext-type query for some time T^*, $\mathcal{O}$ responds by choosing a random $t_0 \in Z_N$ and returning $(w_0^s g_2^{\delta_2} v_0^{t_0} g_2^{\sigma_2 t_0}, g^{t_0} g_2^{t_0}, (u_0^{T^*} h_0)^{t_0} g_2^{t_0(a'T^* + b')})$ as same as Eq 11 to $\mathcal{B}$. Then $\mathcal{B}$ creats the ESF-1 ciphertexts successfully.

When $\mathcal{A}$ requests the secret key of an identity vector $ID|_j = (I_1, \cdots, I_j)$, $\mathcal{B}$ creats the ESF-2-SK key by the HIBE-type query response from $\mathcal{O}$ and the secret key for $ID|_j$ in some node θ is

$$S_{i,0} = g^{\lambda_i} w^{y_i}, S_{i,1} = g^{y_i}, S_{i,2} = g^{r_i}, S_{i,3} = (u^{I_i} h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

$$S_{j,0} = g^{\gamma_\theta - \sum_{i=1}^{j-1} \lambda_i} w^{y_j}, S_{j,1} = g^{y_j}, S_{j,2} = v^{y_j} (X_1X_3)^{r'_j (aI_j + b)} g_2^z, S_{j,3} = (X_1X_3)^{r'_j} g_2^{z'}$$

where $y_1, \cdots, y_j, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_{j-1}, r'_j, z, z' \in Z_n$ are randomly chosen.

When $\mathcal{B}$ creats the IBE private key with the index pair (h, i_c) for some time $\mathcal{T}$ for the identity vector $(I_1, \cdots, I_{j-1})$ in the index h node, the update key with an index pair (h, i_c) is generated as follows:

1. $i_c < h_c$: It randomly chooses $y_0, \cdots, y_{j-1}, \lambda_1, \cdots, \lambda_{j-1}, r'_0, r_1, \cdots, r_{j-1}, z, z' \in Z_n$ and generates a ESF-2-UK $TUK_{ID|_j,T,\theta h}$.

$$U_{0,0} = g^{\alpha-\gamma\theta_h-\sum_{i=1}^{j-1}\lambda_i y_j} w_0^{y_j}, U_{0,1} = g^{y_0}, U_{j,2} = v_0^{y_0}(X_1X_3)^{r'_0(a_0T+b_0)} g_2^z,$$

$$U_{0,3} = (X_1X_3)^{r'_j} g_2^{z'},$$

$$U_{i,0} = g^{\lambda_i} w^{y_i}, U_{i,1} = g^{y_i}, U_{i,2} = g^{r_i}, U_{i,3} = (u^{I_i}h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

It implicitly sets g^{r_0} to be $X_1^{r'_0}$ and that is a properly distribution ESF-2-UK.

2. $i_c = h_c$: $\mathcal{B}$ chooses random values $y_1, \cdots, y_{j-1}, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_{j-1} \in Z_n$. $\mathcal{B}$ forms the challenge key as:

$$U_{0,0} = g^{\alpha-\gamma\theta_h-\sum_{i=1}^{j-1}\lambda_i T_0}, U_{0,1} = T_1, U_{0,2} = T_3, U_{0,3} = T_2$$

$$U_{i,0} = g^{\lambda_i} w^{y_i}, U_{i,1} = g^{y_i}, K_{i,2} = g^{r_i}, U_{i,3} = (u^{I_i}h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

where (T_0, T_1, T_2, T_3) is the challenge IBE key queried to $\mathcal{O}$ who chooses a random $y_0 \in Z_N$ and returns $(T_0, T_1, T_2, T_3) = (w_0^{y_0}, g^{y_0}, v_0^{y_0} T^{a_0 T+b_0}, T)$ to $\mathcal{B}$.

3. $i_c > h_c$: It simply generates a normal IBE private key.

In the challenge IBE key, it implicitly sets g^r to be the G_{p_1} part of T. If $T \in G_{p_1}$, then this matches the distribution of O_0 (since there are no G_{p_3} terms here), and so this will be a properly distributed normal key and $\mathcal{B}$ is playing Game $E_{h_c-1,2}$. If $T \in G_{p_1p3}$, then this matches the distribution of $O_{1/2}$ (note that a, b modulo p_2 are uniformly random and do not occur elsewhere- so there are random G_{p_3} terms attached to the last two group elements) and then $\mathcal{B}$ is playing Game $E_{h_c,1}$.

Hence, if a PPT attacker can distinguish between $E_{h_c-1,2}$ and $E_{h_c,1}$ with non-negligible advantage, $\mathcal{O}$ can distinguish between O_2 and $O_{5/2}$ with non-negligible advantage. It means $\mathcal{O}$ can gain a non-negligible advantage against Assumption 3.

Thus, Under Assumptions 3, no PPT attacker can distinguish between O_2 and $O_{5/2}$ with non-negligible advantage. Thus, no PPT attacker can distinguish between $E_{h_c-1,2}$ and $E_{h_c,1}$ with non-negligible advantage.

Lemma 7 *Under Assumptions 4, no PPT attacker can distinguish between $O_{5/2}$ and O_3 with non-negligible advantage. So no PPT attacker can distinguish between $E_{h_c,1}$ and $E_{h_c,2}$ with non-negligible advantage.*

Proof We assume $\mathcal{B}$ interacts with one of $O_{1/2}$, and O_1. $\mathcal{O}$ receives $g, g_3, X_1 X_2, Y_2 Y_3, T$. $\mathcal{O}$ will simulate either $O_{1/2}$ or O_1 with $\mathcal{B}$, depending on the value of T (which is either in G or G_{p_1p3}). $\mathcal{O}$ picks values $a, b, c, d, a_0, b_0, c_0, d_0 \in Z_N$ uniformly at random and sets $u = g^a$, $h = g^b$, $v = g^c$, $w = g^d$, $u_0 = g^{a_0}$, $h_0 = g^{b_0}$, $v_0 = g^{c_0}$, $w_0 = g^{d_0}$. $\mathcal{B}$ initially obtains the group elements in Eq 84

$$g, u, h, v, w, X_1X_2, w^y(Y_2Y_3)^{y\psi}, g^y(Y_2Y_3)^y, v^y(Y_2Y_3)^{y\sigma},$$

$$u_0, h_0, v_0, w_0, w_0^{y_0}(Y_2Y_3)^{y_0\psi}, g^{y_0}(Y_2Y_3)^{y_0}, v_0^{y_0}(Y_2Y_3)^{y_0\sigma}$$

from its oracle simulator, and gives $\mathcal{A}$ the public parameters in Eq 83.

When $\mathcal{A}$ requests the challenge ciphertext for messages M_0, M_1, identity vector $(I_1^*, \cdots, I_l^*)$ and T^*, $\mathcal{B}$ makes a ciphertext-type query to the oracle for each I_i^* and T^*. When $\mathcal{B}$ makes a ciphertext-type query for some identity I^*, $\mathcal{O}$ responds by choosing a random $t \in Z_N$ and

returning $(w^s g_2^{\delta_1} v^t g_2^{\sigma_1 t}, g^t g_2^t, (u^{I^*} h)^t g_2^{t(a'I^*+b')})$ to $\mathcal{B}$ as same as Eq 10. When $\mathcal{B}$ makes a ciphertext-type query for some time T^*, $\mathcal{O}$ responds by choosing a random $t_0 \in Z_N$ and returning $(w_0^s g_2^{\delta_2} v_0^{t_0} g_2^{\sigma_2 t_0}, g^{t_0} g_2^{t_0}, (u_0^{T^*} h_0)^{t_0} g_2^{t_0(a'T^*+b')})$ as same as Eq 11 to $\mathcal{B}$. Then $\mathcal{B}$ creats the ESF-1 ciphertexts successfully.

When $\mathcal{A}$ requests the secret key of an identity vector $ID|_j = (I_1, \cdots, I_j)$, $\mathcal{B}$ creats the ESF-2-SK key by the HIBE-type query response from $\mathcal{O}$ and the secret key for $ID|_j$ in some node θ is

$$S_{i,0} = g^{\lambda_i} w^{y_i}, S_{i,1} = g^{y_i}, S_{i,2} = g^{r_i}, S_{i,3} = (u^{I_i} h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

$$S_{j,0} = g^{\gamma_\theta - \sum_{i=1}^{j-1} \lambda_i} w^{y_j}, S_{j,1} = g^{y_j}, S_{j,2} = v^{y_j} (X_1 X_2)^{r'_j (aI_j + b)} g_3^z, S_{j,3} = (X_1 X_2)^{r'_j} g_3^{z'}$$

where $y_1, \cdots, y_j, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_{j-1}, r'_j, z, z' \in Z_n$ are randomly chosen.

When $\mathcal{B}$ creats the IBE private key with the index pair (h, i_c) for a time T in the index h node in the binary tree $BT_{ID|_j = (I1, \cdots, Ij-1)}$, the update key with an index pair (h, i_c) is generated as follows:

1. $i_c < h_c$: $\mathcal{B}$ chooses random values $y_1, \cdots, y_{j-1}, \lambda_1, \cdots, \lambda_{j-1}, r_0, \cdots, r_{j-1}, z, z' \in Z_n$ and generates a ESF-2 $TUK_{IBE,h}$.

$$U_{0,0} = g^{\gamma_\theta - \sum_{i=1}^{j-1} \lambda_i} (w_0^{y_0}, U_{0,1} = g^{y_0}, U_{0,2} = v_0^{y_0} (u_0^T h_0)^{r_0} (Y_2 Y_3)^z, U_{0,3} = g^{r_0} (Y_2 Y_3)^{z'}$$

$$U_{i,0} = g^{\lambda_i} w^{y_i}, U_{i,1} = g^{y_i}, K_{i,2} = g^{r_i}, U_{i,3} = (u^{I_i} h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

2. $i_c = h_c$: $\mathcal{B}$ chooses random values $y_1, \cdots, y_{j-1}, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_{j-1} \in Z_n$. $\mathcal{B}$ forms the challenge key as:

$$U_{0,0} = g^{\alpha - \gamma_{\theta_h} - \sum_{i=1}^{j-1} \lambda_i} T_0, U_{0,1} = T_1, U_{0,2} = T_3, U_{0,3} = T_2$$

$$U_{i,0} = g^{\lambda_i} w^{y_i}, U_{i,1} = g^{y_i}, U_{i,2} = g^{r_i}, U_{i,3} = (u^{I_i} h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

where (T_0, T_1, T_2, T_3) is the challenge IBE key queried to $\mathcal{O}$ who chooses a random $y_0 \in Z_N$ and returns $(T_0, T_1, T_2, T_3) = (w_0^{y_0}, g^{y_0}, v^{y_0} T^{a_0 T + b_0}, T)$ to $\mathcal{B}$.

3. $i_c > h_c$: It simply generates a normal HIBE private key.

As in the previous lemma, this implicitly sets g_{r_0} to be the G_{p_1} part of T in the challenge IBE key. We note that a_0, b_0 modulo p_2, p_3 are uniformly random and do not appear elsewhere. Thus, when $T \in G_{p_1 p 3}$, these last two terms will have random elements of G_{p_3} attached (matching the distribution of $O_{5/2}$) and then $\mathcal{B}$ is playing Game $E_{h_c,1}$. And when $T \in G$, these last two terms will have random elements in both G_{p_3} and G_{p_2} attached (matching the distribution of O_3) and then $\mathcal{B}$ is playing Game $E_{h_c,2}$.

Hence, if a PPT attacker can distinguish between $E_{h_c,1}$ and $E_{h_c,2}$ with non-negligible advantage, $\mathcal{O}$ can distinguish between $O_{5/2}$ and O_3 with non-negligible advantage. It means $\mathcal{O}$ can gain a non-negligible advantage against Assumption 4.

Thus, Under Assumptions 4, no PPT attacker can distinguish between $O_{5/2}$ and O_3 with non-negligible advantage. Thus, no PPT attacker can distinguish between $E_{h_c,1}$ and $E_{h_c,2}$ with non-negligible advantage.

Lemma 8 *Under Assumptions 3, no PPT attacker can distinguish between O_3 and $O_{3.1}$ with non-negligible advantage. So no PPT attacker can distinguish between F_{h_c-1} and F_{h_c} with non-negligible advantage.*

Proof We assume $\mathcal{B}$ interacts with one of O_3, $O_{3.1}$. $\mathcal{O}$ receives $g, g_2, X_1 X_3, Y_1 Y_3, T$. $\mathcal{O}$ will simulate either O_0 or $O_{1/2}$ with $\mathcal{B}$, depending on the value of T (which is either in G_{p_1} or $G_{p_1 p3}$). $\mathcal{O}$ picks values $a, b, c, d, a_0, b_0, c_0, d_0 \in Z_N$ uniformly at random and sets $u = g^a$, $h = g^b$, $v = g^c$, $w = g^d$, $u_0 = g^{a_0}$, $h_0 = g^{b_0}$, $v_0 = g^{c_0}$, $w_0 = g^{d_0}$. $\mathcal{B}$ initially obtains the group elements in Eq 82

$$g, u, h, v, w, g^s g_2^{\gamma}, (X_1X_3)^d g_2^{y_2 d}, (X_1X_3) g_2^{y_2}, (X_1X_3)^c g_2^{cy_2}$$

$$u_0, h_0, v_0, w_0, (X_1X_3)^{zd_0} g_2^{y'_2 d_0}, (X_1X_3)^z g_2^{y'_2}, (X_1X_3)^{zc_0} g_2^{c_0 y'_2}$$

from its oracle simulator who additionally chooses $s, \gamma, \delta_1, \delta_2, y_2, y'_2 \in Z_N$ randomly. $\mathcal{B}$ chooses $\alpha \in Z_n$ randomly, and gives $\mathcal{A}$ the following public parameters in Eq 83. We note that $\mathcal{B}$ knows the master secret key α.

When $\mathcal{A}$ requests the challenge ciphertext for messages M_0, M_1, identity vector $(I_1^*, \ldots, I_l^*)$ and T^*, $\mathcal{B}$ makes a ciphertext-type query to the oracle for each I_i^* and T^*. When $\mathcal{B}$ makes a ciphertext-type query for some identity I^*, $\mathcal{O}$ responds by choosing a random $t \in Z_N$ and returning $(w^s g_2^{\delta_1} v^t g_2^{ct}, g^t g_2^t, (u^{I^*} h)^t g_2^{a' I^* + b'})$ to $\mathcal{B}$. When $\mathcal{B}$ makes a ciphertext-type query for some time T^*, $\mathcal{O}$ responds by choosing a random $t_0 \in Z_N$ and returning $(w_0^s g_2^{\delta_2} v_0^{t_0} g_2^{c_0 t_0}, g^{t_0} g_2^{t_0}, (u_0^{T^*} h_0)^{t_0} g_2^{a' T^* + b'})$ to $\mathcal{B}$. Then $\mathcal{B}$ creats the ESF-1 ciphertexts successfully.

Upon receiving a challenge IBE-key-type query for $T \in Z_n$, $\mathcal{O}$ chooses $r_1, r_2, r', y'' \in Z_n$ randomly and returns the group elements

$$((g)^{d_0 y''}, (g)^{y''}, (g)^{c_0 y''} (X_1X_3)^{(a_0 T + b_0) r'} g_2^{r_1}, (X_1X_3)^{r'} g_2^{r_2})$$

to $\mathcal{B}$. And then $\mathcal{B}$ creats the ESF-2 update key by using the group elements.

When $\mathcal{B}$ creats the HIBE private key with the index pair (h, i_c) for some identity vector $(I_1, \cdots, I_j)$ in the index h node, the HIBE private key with an index pair (h, i_c) is generated as follows:

1. $i_c < h_c$: It randomly chooses $y_1, \cdots, y_{j-1}, y'_j, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_{j-1}, r'_j, z, z' \in Z_n$ and generates a ESF-3-SK $PSK_{HIBE,h}$.

$$K_{i,0} = g^{\lambda_i} w^{y_i}, K_{i,1} = g^{y_i}, K_{i,2} = g^{r_i}, K_{i,3} = (u^{I_i} h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

$$K_{j,0} = g^{\gamma_\theta - \sum_{i=1}^{j-1} \lambda_i} (X_1X_3)^{dy'_j}, K_{j,1} = (X_1X_3)^{y'_j},$$

$$K_{j,2} = (X_1X_3)^{cy'_j} (X_1X_3)^{r'_j (aI_j + b)} g_2^z, K_{j,3} = (X_1X_3)^{r'_j} g_2^{z'}$$

   It implicitly sets g^{y_j} to be $X_1^{y'_j}$ and g^{r_j} to be $X_1^{r'_j}$ and that is a properly distribution ESF-3-SK.

2. $i_c = h_c$: $\mathcal{B}$ chooses random values $y_1, \cdots, y_{j-1}, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_{j-1} \in Z_n$. $\mathcal{B}$ forms the challenge key as:

$$K_{i,0} = g^{\lambda_i} w^{y_i}, K_{i,1} = g^{y_i}, K_{i,2} = g^{r_i}, K_{i,3} = (u^{I_i} h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

$$K_{j,0} = g^{\gamma_\theta - \sum_{i=1}^{j-1} \lambda_i} T_0, K_{j,1} = T_1, K_{j,2} = T_3, K_{j,3} = T_2$$

   where (T_0, T_1, T_2, T_3) is the challenge HIBE key queried to $\mathcal{O}$ who chooses a random $r, r_1, r_2 \in Z_N$ and returns $(T_0, T_1, T_2, T_3) = (T^d, T, T^c (X_1X_3)^{r(aI+b)} g_2^{r_1}, (X_1X_3)^r g_2^{r_2})$ to $\mathcal{B}$.

3. $i_c > h_c$: It randomly chooses $y_1, \cdots, y_j, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_{j-1}, r'_j, z, z' \in Z_n$ and generates a ESF-2-SK $PSK_{HIBE,h}$.

$$K_{i,0} = g^{\lambda_i} w^{y_i}, K_{i,1} = g^{y_i}, K_{i,2} = g^{r_i}, K_{i,3} = (u^{I_i} h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

$$K_{j,0} = g^{\gamma_\theta - \sum_{i=1}^{j-1} \lambda_i} w^{y_j}, K_{j,1} = g^{y_j}, K_{j,2} = v^{y_j} (X_1 X_3)^{r'_j (aI_j + b)} g_2^z, K_{j,3} = (X_1 X_3)^{r'_j} g_2^{z'}$$

It implicitly sets g^{r_j} to be $X_1^{r'_j}$ and that is a properly distribution ESF-2-SK.

In the challenge HIBE key, it implicitly sets $g^{y'}$ to be the G_{p_1} part of T. If $T \in G_{p_1}$, then this matches the distribution of O_3 (since there are no G_{p_3} terms here), and so this will be a properly distributed normal key and $\mathcal{B}$ is playing Game F_{h_c-1}. If $T \in G_{p_1 p 3}$, then this matches the distribution of $O_{3.1}$ (note that a, b modulo p_2 are uniformly random and do not occur elsewhere-so there are random G_{p_3} terms attached to the last two group elements) and then $\mathcal{B}$ is playing Game F_{h_c}.

Hence, if a PPT attacker can distinguish between F_{h_c-1} and F_{h_c} with non-negligible advantage, $\mathcal{O}$ can distinguish between O_3 and $O_{3.1}$ with non-negligible advantage. It means $\mathcal{O}$ can gain a non-negligible advantage against Assumption 3.

Thus, Under Assumptions 3, no PPT attacker can distinguish between O_3 and $O_{3.1}$ with non-negligible advantage. Thus, no PPT attacker can distinguish between F_{h_c-1} and F_{h_c} with non-negligible advantage.

Lemma 9 *Under Assumptions 3, no PPT attacker can distinguish between $O_{3.1}$ and $O_{3.2}$ with non-negligible advantage. So no PPT attacker can distinguish between $F1_{h_c-1}$ and $F1_{h_c}$ with non-negligible advantage.*

Proof The proof of this lemma is almost the same as that of Lemma 8 except the generation of secret keys and update keys.

Upon receiving a challenge HIBE-key-type query for $I \in Z_n$, $\mathcal{O}$ chooses $r_1, r_2, r', y'' \in Z_n$ randomly and returns the group elements

$$((X_1 X_3)^{dy''}, (X_1 X_3)^{y''}, (X_1 X_3)^{cy''} (X_1 X_3)^{(a_I + b) r'} g_2^{r_1}, (X_1 X_3)^{r'} g_2^{r_2})$$

to $\mathcal{B}$. And then $\mathcal{B}$ creats the ESF-3 update key by using the group elements.

When $\mathcal{B}$ creats the IBE private key with the index pair (h, i_c) for some identity vector $(I_1, \cdots, I_{j-1})$ and the time T in the index h node, the IBE private key with an index pair (h, i_c) is generated as follows:

1. $i_c < h_c$: It randomly chooses $y_1, \cdots, y_{j-1}, y'_0, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_{j-1}, r'_0, z, z' \in Z_n$ and generates a ESF-3-UK $EUK_{IBE,h}$.

$$U_{0,0} = g^{\alpha - \gamma_\theta - \sum_{i=1}^{j-1} \lambda_i} (X_1 X_3)^{d_0 y'_0}, U_{0,1} = (X_1 X_3)^{y'_0},$$

$$U_{0,2} = (X_1 X_3)^{c_0 y'_0} (X_1 X_3)^{r'_0 (a_0 T + b_0)} g_2^z, U_{0,3} = (X_1 X_3)^{r'_0} g_2^{z'}$$

$$U_{i,0} = g^{\lambda_i} w^{y_i}, U_{i,1} = g^{y_i}, U_{i,2} = g^{r_i}, U_{i,3} = (u^{I_i} h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

It implicitly sets g^{y_0} to be $X_1^{y'_0}$ and g^{r_0} to be $X_1^{r'_0}$ and that is a properly distribution ESF-3-UK.

2. $i_c = h_c$: $\mathcal{B}$ chooses random values $y_1, \cdots, y_{j-1}, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_{j-1} \in Z_n$. $\mathcal{B}$ forms the challenge key as:

$$U_{0,0} = g^{\alpha-\gamma_\theta-\sum_{i=1}^{j-1}\lambda_i} T_0,$$

$$U_{0,1} = T_1, U_{0,2} = T_3, U_{0,3} = T_2$$

$$U_{i,0} = g^{\lambda_i} w^{y_i}, U_{i,1} = g^{y_i}, U_{i,2} = g^{r_i}, U_{i,3} = (u^{I_i} h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

where (T_0, T_1, T_2, T_3) is the challenge IBE key queried to $\mathcal{O}$ who chooses a random r, r_1, $r_2 \in Z_N$ and returns $(T_0, T_1, T_2, T_3) = (T^{d_0}, T, T^{c_0}(X_1X_3)^{r(a_0T+b_0)} g_2^{r_1}, (X_1X_3)^r g_2^{r_2})$ to $\mathcal{B}$.

3. $i_c > h_c$: It randomly chooses $y_1, \cdots, y_j, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_{j-1}, r'_j, z, z' \in Z_n$ and generates a ESF-2-UK $EUK_{IBE,h}$.

$$U_{0,0} = g^{\gamma_\theta-\sum_{i=1}^{j-1}\lambda_i} w_{)}^{y_0}, U_{0,1} = g^{y_0}, U_{0,2} = v_0^{y_0}(X_1X_3)^{r'_0(a_0T+b_0)} g_2^z, U_{0,3} = (X_1X_3)^{r'_j} g_2^{z'}$$

$$U_{i,0} = g^{\lambda_i} w^{y_i}, U_{i,1} = g^{y_i}, U_{i,2} = g^{r_i}, U_{i,3} = (u^{I_i} h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

It implicitly sets g^{r_0} to be $X_1^{r'_0}$ and that is a properly distribution ESF-2-UK.

In the challenge IBE key, it implicitly sets $g^{y'}$ to be the G_{p_1} part of T. If $T \in G_{p_1}$, then this matches the distribution of $O_{3.1}$ (since there are no G_{p_3} terms here), and so this will be a properly distributed normal key and $\mathcal{B}$ is playing Game $F1_{h_c-1}$. If $T \in G_{p_1p3}$, then this matches the distribution of $O_{3.2}$ (note that a, b modulo p_2 are uniformly random and do not occur elsewhere- so there are random G_{p_3} terms attached to the last two group elements) and then $\mathcal{B}$ is playing Game $F1_{h_c}$.

Hence, if a PPT attacker can distinguish between $F1_{h_c-1}$ and $F1_{h_c}$ with non-negligible advantage, $\mathcal{O}$ can distinguish between $O_{3.1}$ and $O_{3.2}$ with non-negligible advantage. It means $\mathcal{O}$ can gain a non-negligible advantage against Assumption 3.

Thus, Under Assumptions 3, no PPT attacker can distinguish between $O_{3.1}$ and $O_{3.2}$ with non-negligible advantage. Thus, no PPT attacker can distinguish between $F1_{h_c-1}$ and $F1_{h_c}$ with non-negligible advantage.

Lemma 10 *Under Assumptions 4, no PPT attacker can distinguish between $O_{3.2}$ and $O_{3.3}$ with non-negligible advantage. So no PPT attacker can distinguish between $G_{ESF'-2}$ and $G_{ESF'-3}$ with non-negligible advantage.*

Proof We assume $\mathcal{B}$ interacts with one of $O_{3.2}$, $O_{3.3}$. $\mathcal{O}$ receives $g, g_2, X_1 X_3, Y_2 Y_3, T$. $\mathcal{O}$ will simulate either $O_{3.2}$ or $O_{3.3}$ with $\mathcal{B}$, depending on the value of T (which is either in G or G_{p_1p2}). $\mathcal{O}$ picks values $a, b, c, d, a_0, b_0, c_0, d_0 \in Z_N$ uniformly at random and sets $u = g^a$, $h = g^b$, $v = g^c$, $w = g^d$, $u_0 = g^{a_0}$, $h_0 = g^{b_0}$, $v_0 = g^{c_0}$, $w_0 = g^{d_0}$. It chooses random values $\sigma_1, \sigma_2, y, y', t_3, z \in Z_N$ and then $\mathcal{B}$ initially obtains the group elements

$$\begin{aligned} &g, u, h, v, w, T, w^y(Y_2Y_3)^d, g^y(Y_2Y_3), v^y(Y_2Y_3)^{\sigma_1} \\ &u_0, h_0, v_0, w_0, w_0^{y'}(Y_2Y_3)^{zd_0}, g^{y'}(Y_2Y_3)^z, v_0^{y'}(Y_2Y_3)^{\sigma_2} \end{aligned} \tag{86}$$

We note that this sets $\psi_1 = d$ modulo p_2 and p_3. It implicitly sets g^s to be the G_{p_1} part of T. If $T \in G_{p_1p2}$, this is distributed identically to the initial elements provided by $O_{3.2}$. If $T \in G$, this is distributed identically to the initial elements provided by $O_{3.3}$.

Upon receiving a challenge IBE-key-type query for $T \in Z_n$, $\mathcal{O}$ chooses r_1, r_2, r', $y'' \in Z_n$ randomly and returns the group elements

$$((g)^{d_0 y''}, (g)^{y''}, (g)^{c_0 y''} (X_1 X_3)^{(a_0 T + b_0) r'} g_2^{r_1}, (X_1 X_3)^{r'} g_2^{r_2})$$

to $\mathcal{B}$. And then $\mathcal{B}$ creats the ESF-2 update key by using the group elements.

When $\mathcal{B}$ creats the HIBE private key with the index pair (h, i_c) for some identity vector $(I_1, \cdots, I_j)$ in the index h node, the HIBE private key with an index pair (h, i_c) is generated as follows:

When $\mathcal{A}$ requests the challenge ciphertext for messages M_0, M_1, identity vector $(I_1^*, \cdots, I_l^*)$ and T^*, $\mathcal{B}$ makes a ciphertext-type query to the oracle for each I_i^* and T^*. When $\mathcal{B}$ makes a ciphertext-type query for some identity I^*, $\mathcal{O}$ responds by choosing a random $t \in Z_N$ and returning $(w^s g_2^{\delta_1} v^t g_2^{ct}, g^t g_2^t, (u^{I^*} h)^t g_2^{a' I^* + b'})$ to $\mathcal{B}$. When $\mathcal{B}$ makes a ciphertext-type query for some time T^*, $\mathcal{O}$ responds by choosing a random $t_0 \in Z_N$ and returning $(w_0^s g_2^{\delta_2} v_0^{t_0} g_2^{c_0 t_0}, g^{t_0} g_2^{t_0}, (u_0^{T^*} h_0)^{t_0} g_2^{a' T^* + b'})$ to $\mathcal{B}$. Then $\mathcal{B}$ creats the ESF-1 ciphertexts successfully.

Lemma 11 *Under Assumptions 4, no PPT attacker can distinguish between $O_{3.3}$ and $O_{3.4}$ with non-negligible advantage. So no PPT attacker can distinguish between $G_{ESF'-3}$ and $G_{ESF'-4}$ with non-negligible advantage.*

Proof We assume $\mathcal{B}$ interacts with one of $O_{3.3}$, $O_{3.4}$. $\mathcal{O}$ receives g, g_3, X_1 X_2, Y_2 Y_3, T. $\mathcal{O}$ will simulate either $O_{3.3}$ or $O_{3.4}$ with $\mathcal{B}$, depending on the value of T (which is either in G or $G_{p_1 p3}$). $\mathcal{O}$ picks values a, b, c, d, a_0, b_0, c_0, $d_0 \in Z_N$ uniformly at random and sets $u = g^a$, $h = g^b$, $v = g^c$, $w = g^d$, $u_0 = g^{a_0}$, $h_0 = g^{b_0}$, $v_0 = g^{c_0}$, $w_0 = g^{d_0}$. $\mathcal{B}$ initially obtains the group elements

$$\begin{gathered} g, u, h, v, w, g^s (Y_2 Y_3)^{\gamma}, w^y (Y_2 Y_3)^d, g^y Y_2 Y_3, v^y Y_2 Y_3{}^c \\ u_0, h_0, v_0, w_0, w_0^{y'} (Y_2 Y_3)^{z d_0}, g^{y'} Y_2 Y_3{}^z, v_0^{y'} Y_2 Y_3{}^{z c_0} \end{gathered} \tag{87}$$

from its oracle simulator who additionally chooses s, γ, y, y', $z \in Z_N$ randomly. We note that this is properly distributed and set $\psi_1 = d$ modulo p_2 and p_3, $\psi_2 = d_0$ modulo p_2 and p_3 and $\sigma_1 = c$ modulo p_2 and p_3, $\sigma_2 = c_0$ modulo p_2 and p_3.

When $\mathcal{A}$ requests the challenge ciphertext for messages M_0, M_1, identity vector $(I_1^*, \cdots, I_l^*)$ and T^*, $\mathcal{B}$ makes a ciphertext-type query to the oracle for each I_i^* and T^*. When $\mathcal{B}$ makes a ciphertext-type query for some identity I^*, $\mathcal{O}$ responds by choosing a random $t \in Z_N$ and returning $(w^s (Y_2 Y_3)^{\delta_1} (X_1 X_2)^{ct'}, (X_1 X_2)^{t'} g^{t_3}, (X_1 X_2)^{a I^* + b} g_3^{t'_3})$ to $\mathcal{B}$. When $\mathcal{B}$ makes a ciphertext-type query for some time T^*, $\mathcal{O}$ responds by choosing a random $t_0 \in Z_N$ and returning $(w_0^s (Y_2 Y_3)^{\delta_2} (X_1 X_2)^{c_0 t'}, (X_1 X_2)^{t'} g^{t_3}, (X_1 X_2)^{a_0 T^* + b_0} g_3^{t'_3})$ to $\mathcal{B}$. This implicitly sets $g^t = X_1^{t'}$. It also sets $a' = a$ and $b' = b$ modulo p_2 or $a' = a_0$ and $b' = b_0$ modulo p_2,

which are properly distributed because a, b modulo p_2 and a_0, b_0 modulo p_2 do not appear elsewhere. Then $\mathcal{B}$ creats the ESF-5 ciphertexts successfully.

Upon receiving a challenge IBE-key-type query for $T \in Z_n$, $\mathcal{O}$ chooses r, y', $z \in Z_n$ randomly and returns the group elements

$$(w_0^{y'} g_3^{y' \psi}, g^{y'} g_3^{y'}, g^r (Y_2 Y_3), v_0^{y'} (u_0^T h_0)^r (Y_2 Y_3)^z)$$

to $\mathcal{B}$. And then $\mathcal{B}$ creats the ESF-3 update key by using the group elements.

When $\mathcal{B}$ creats the HIBE private key with the index pair (h, i_c) for some identity vector $(I_1, \cdots, I_j)$ in the index h node, the HIBE private key with an index pair (h, i_c) is generated as follows:

1. $i_c < h_c$: It randomly chooses $y_1, \cdots, y_j, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_j, z, z' \in Z_n$ and generates a ESF-4-SK $PSK_{HIBE,h}$.

$$K_{i,0} = g^{\lambda_i} w^{y_i}, K_{i,1} = g^{y_i}, K_{i,2} = g^{r_i}, K_{i,3} = (u^{I_i} h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

$$K_{j,0} = g^{\gamma_\theta - \sum_{i=1}^{j-1} \lambda_i} w^{y_j} (Y_2 Y_3)^{y_j \psi_1}, K_{j,1} = g^{y_j} (Y_2 Y_3)^{y_j},$$

$$K_{j,2} = v^{y_j} (u^{I_j} h)^{r_j} (Y_2 Y_3)^{z}, K_{j,3} = g^{r_j} (Y_2 Y_3)^{z'}$$

That is a properly distribution ESF-4-SK.

2. $i_c = h_c$: $\mathcal{B}$ chooses random values $y_1, \cdots, y_{j-1}, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_{j-1} \in Z_n$. $\mathcal{B}$ forms the challenge key as:

$$K_{i,0} = g^{\lambda_i} w^{y_i}, K_{i,1} = g^{y_i}, K_{i,2} = g^{r_i}, K_{i,3} = (u^{I_i} h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

$$K_{j,0} = g^{\gamma_\theta - \sum_{i=1}^{j-1} \lambda_i} T_0, K_{j,1} = T_1, K_{j,2} = T_3, K_{j,3} = T_2$$

where (T_0, T_1, T_2, T_3) is the challenge HIBE key queried to $\mathcal{O}$ who chooses a random r, r_1, $r_2 \in Z_N$ and returns $(T_0, T_1, T_2, T_3) = (T^d, T, T^c (u^{I_j} h)^r (Y_2 Y_3)^{r_1}, g^r (Y_2 Y_3)^{r_2})$ to $\mathcal{B}$.

3. $i_c > h_c$: It randomly chooses $y_1, \cdots, y_j, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_j, z, z' \in Z_n$ and generates a ESF-3-SK $PSK_{HIBE,h}$.

$$K_{i,0} = g^{\lambda_i} w^{y_i}, K_{i,1} = g^{y_i}, K_{i,2} = g^{r_i}, K_{i,3} = (u^{I_i} h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

$$K_{j,0} = g^{\gamma_\theta - \sum_{i=1}^{j-1} \lambda_i} w^{y_j} g_3^{y_j \psi_1}, K_{j,1} = g^{y_j} g_3^{y_j},$$

$$K_{j,2} = v^{y_j} (u^{I_j} h)^{r_j} (Y_2 Y_3)^{z}, K_{j,3} = g^{r_j} (Y_2 Y_3)^{z'}$$

That is a properly distribution ESF-3-SK.

In the challenge HIBE key, it implicitly sets $g^{y'}$ to be the G_{p_1} part of T. If $T \in G_{p_2 p 3}$, then this matches the distribution of $O_{3.3}$, and so this will be a properly distributed normal key and $\mathcal{B}$ is playing Game $F3_{h_c-1}$. If $T \in G$, then this matches the distribution of $O_{3.4}$ and then $\mathcal{B}$ is playing Game $F3_{h_c}$.

Hence, if a PPT attacker can distinguish between $F3_{h_c-1}$ and $F3_{h_c}$ with non-negligible advantage, $\mathcal{O}$ can distinguish between $O_{3.3}$ and $O_{3.4}$ with non-negligible advantage. It means $\mathcal{O}$ can gain a non-negligible advantage against Assumption 4.

Thus, Under Assumptions 4, no PPT attacker can distinguish between $O_{3.3}$ and $O_{3.4}$ with non-negligible advantage. Thus, no PPT attacker can distinguish between $F3_{h_c-1}$ and $F3_{h_c}$ with non-negligible advantage.

Lemma 12 *Under Assumptions 4, no PPT attacker can distinguish between $O_{3.4}$ and $O_{3.5}$ with non-negligible advantage. So no PPT attacker can distinguish between $G_{ESF'-4}$ and $G_{ESF'-5}$ with non-negligible advantage.*

Proof The proof of this lemma is almost the same as that of Lemma 11 except the generation of secret keys and update leys.

Upon receiving a challenge HIBE-key-type query for $I \in Z_n$, $\mathcal{O}$ chooses $r, y', z, z' \in Z_n$ randomly and returns the group elements

$$(w^{y'} (Y_2 Y_3)^{y' \psi}, g^{y'} (Y_2 Y_3)^{y'}, g^r (Y_2 Y_3)^z, v^{y'} (u^I h)^r (Y_2 Y_3)^{z'})$$

to $\mathcal{B}$. And then $\mathcal{B}$ creats the ESF-4 secret key by using the group elements.

When $\mathcal{B}$ creats the IBE private key with the index pair (h, i_c) for some time $\mathcal{T}$ for the identity vector $(I_1, \cdots, I_{j-1})$ in the index h node, the update key with an index pair (h, i_c) is generated as follows:

1. $i_c < h_c$: It randomly chooses $y_1, \cdots, y_j, \lambda_1, \cdots, \lambda_{j-1}, r_0, \cdots, r_{j-1}, z, z' \in Z_n$ and generates a ESF-4-UK $TUK_{IBE,h}$.

$$U_{0,0} = g^{\alpha-\gamma_\theta-\sum_{i=1}^{j-1}\lambda_i} w_0^{y_0}(Y_2Y_3)^{y_0\psi_2}, U_{0,1} = g^{y_0}(Y_2Y_3)^{y_0},$$

$$U_{0,2} = v_0^{y_0}(u_0^T h_0)^{r_0}(Y_2Y_3)^z, U_{0,3} = g^{r_0}(Y_2Y_3)^{z'},$$

$$U_{i,0} = g^{\lambda_i} w^{y_i}, U_{i,1} = g^{y_i}, U_{i,2} = g^{r_i}, U_{i,3} = (u^{I_i}h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

That is a properly distribution ESF-4-UK.

2. $i_c = h_c$: $\mathcal{B}$ chooses random values $y_1, \cdots, y_{j-1}, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_{j-1} \in Z_n$. $\mathcal{B}$ forms the challenge key as:

$$U_{0,0} = g^{\alpha-\gamma_\theta-\sum_{i=1}^{j-1}\lambda_i} T_0, U_{0,1} = T_1, U_{0,2} = T_3, U_{0,3} = T_2$$

$$U_{i,0} = g^{\lambda_i} w^{y_i}, U_{i,1} = g^{y_i}, U_{i,2} = g^{r_i}, U_{i,3} = (u^{I_j}h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

where (T_0, T_1, T_2, T_3) is the challenge IBE key queried to $\mathcal{O}$ who chooses a random $r, r_1, r_2 \in Z_N$ and returns $(T_0, T_1, T_2, T_3) = (T^{d_0}, T, T^{c_0}(u_0^T h_0)^r (Y_2Y_3)^{r_1}, g^r (Y_2Y_3)^{r_2})$ to $\mathcal{B}$.

3. $i_c > h_c$: It randomly chooses $y_0, y_1, \cdots, y_{j-1}, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_j, z, z' \in Z_n$ and generates a ESF-3-UK $TUK_{IBE,h}$.

$$U_{0,0} = g^{\alpha-\gamma_\theta-\sum_{i=1}^{j-1}\lambda_i} w_0^{y_0} g_3^{y_0\psi_1}, U_{0,1} = g^{y_0} g_3^{y_0}, U_{0,2} = v_0^{y_0}(u_0^T h_0)^{r_0}(Y_2Y_3)^z,$$

$$U_{0,3} = g^{r_j}(Y_2Y_3)^{z'}$$

$$U_{i,0} = g^{\lambda_i} w^{y_i}, U_{i,1} = g^{y_i}, U_{i,2} = g^{r_i}, U_{i,3} = (u^{I_i}h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

That is a properly distribution ESF-3-UK.

In the challenge IBE key, it implicitly sets $g^{y'}$ to be the G_{p_1} part of T. If $T \in G_{p_2p3}$, then this matches the distribution of $O_{3.4}$, and so this will be a properly distributed normal key and $\mathcal{B}$ is playing Game $F4_{h_c-1}$. If $T \in G$, then this matches the distribution of $O_{3.5}$ and then $\mathcal{B}$ is playing Game $F4_{h_c}$.

Hence, if a PPT attacker can distinguish between $F4_{h_c-1}$ and $F4_{h_c}$ with non-negligible advantage, $\mathcal{O}$ can distinguish between $O_{3.4}$ and $O_{3.5}$ with non-negligible advantage. It means $\mathcal{O}$ can gain a non-negligible advantage against Assumption 4.

Thus, Under Assumptions 4, no PPT attacker can distinguish between $O_{3.4}$ and $O_{3.5}$ with non-negligible advantage. Thus, no PPT attacker can distinguish between $F4_{h_c-1}$ and $F4_{h_c}$ with non-negligible advantage.

Lemma 13 *Under Assumptions 4, no PPT attacker can distinguish between $O_{3.5}$ and $O_{3.6}$ with non-negligible advantage. So no PPT attacker can distinguish between G_{ESF-5} and G_{ESF-6} with non-negligible advantage.*

Proof We assume $\mathcal{B}$ interacts with one of $O_{3.5}$, $O_{3.6}$. $\mathcal{O}$ receives $g, g_2, X_1 X_3, Y_2 Y_3, T$. $\mathcal{O}$ will simulate either $O_{3.5}$ or $O_{3.6}$ with $\mathcal{B}$, depending on the value of T (which is either in G or G_{p_1p2}). $\mathcal{O}$ picks values $a, b, c, d, a_0, b_0, c_0, d_0 \in Z_N$ uniformly at random and sets $u = g^a$, $h = g^b$, $v = g^c$,

$w = g^d$, $u_0 = g^{a_0}$, $h_0 = g^{b_0}$, $v_0 = g^{c_0}$, $w_0 = g^{d_0}$. $\mathcal{B}$ initially obtains the group elements

$$g, u, v, w, T, w^y (Y_2 Y_3)^d, g^y (Y_2 Y_3), v^y (Y_2 Y_3)_1^\sigma,$$
$$u_0, v_0, w_0, w_0^{y_0} (Y_2 Y_3)^{d_0}, g^{y_0} (Y_2 Y_3)^z, v_0^{y_0} (Y_2 Y_3)^{\sigma_2} \tag{88}$$

from its oracle simulator where $z, y_0, y, \sigma_1, \sigma_2 \in Z_p$ are randomly chosen. We note that this set $\psi = d$ modulo p_2 and p_3. If $T \in G_{p_1 p 2}$, then this matches the initial elements provided by $O_{3.6}$. If $T \in G$, then this matches the initial elements provided by $O_{3.5}$. It chooses $\alpha \in Z_n$ randomly, and gives $\mathcal{A}$ the following public parameters in Eq 83. We note that $\mathcal{B}$ knows the master secret key α. When $\mathcal{A}$ requests a normal update key or a normal decryption key, $\mathcal{B}$ can responds by using the usual key generation algorithm, since it knows α.

When $\mathcal{A}$ makes a secret key query for the identity $ID|_j = (I_1, \cdots, I_j)$, then $\mathcal{B}$ makes its challenge HIBE-key-type query for I_j, $\mathcal{O}$ responds as follows. It chooses $y', r, r_1, r_2 \in Z_N$ randomly and responds with:

$$((X_1 X_3 g_2)^{dy'}, (X_1 X_3 g_2)^{y'}, (X_1 X_3 g_2)^{cy'} (u^{I_j} h)^r (Y_2 Y_3)^{r_1}, g^r (Y_2 Y_3)^{r_2})$$

And then $\mathcal{B}$ creats the ESF-4 secret key by using the group elements.

Upon receiving a challenge IBE-key-type query for $T \in Z_n$, $\mathcal{O}$ chooses $y', r, r_1, r_2 \in Z_N$ randomly and responds with:

$$((X_1 X_3 g_2)^{d_0 y'}, (X_1 X_3 g_2)^{y'}, (X_1 X_3 g_2)^{c_0 y'} (u_0^T h_0)^r (Y_2 Y_3)^{r_1}, g^r (Y_2 Y_3)^{r_2})$$

to $\mathcal{B}$. And then $\mathcal{B}$ creats the ESF-4 update key by using the group elements.

When $\mathcal{A}$ requests the challenge ciphertext for messages M_0, M_1, identity vector $(I_1^*, \cdots, I_l^*)$ and T^*, $\mathcal{B}$ makes a ciphertext-type query to the oracle for each I_i^* and T^*. In response to each query for I_i^* or T^*, $\mathcal{O}$ gets random $t_0, t_1, \ldots, t_l \in Z_p$, chooses $\beta \in \{0, 1\}$ and creats the ciphertext as

$$(C = M_\beta e(X_1 X_3, g)^\alpha, C_0 = X_1 X_3, \{C_{i,1}, C_{i,2}, C_{i,3}\}_{i=0}^l)$$

where the ciphertext-element-group $(C_{i,1}, C_{i,2}, C_{i,3})$ is defined as follows if $i > 0$:

$$(T^d T^{ct_3} v^{t_i} g_2^{\sigma_1 t_i}, T^{t_3} g^{t_i} g_2^{t_i}, T^{t_3 (aI_i^* + b)} g_2^{t_i (aI_i^* + b)})$$

and $(C_{0,1}, C_{0,2}, C_{0,3})$ is defined as follows

$$(T^{d_0} T^{c_0 t_3} v_0^{t_0} g_2^{\sigma_2 t_0}, T^{t_3} g^{t_0} g_2^{t_0}, T^{t_3 (a_0 T^* + b_0)} g_2^{t_0 (a_0 T^* + b_0)})$$

We note that this is very similar to the way $\mathcal{O}$ behaves in the proof of Lemma 12. The only difference is the $g_2^{dy'}, g_2^{y'}, g_2^{cy'}$ terms which have been added to the challenge key. As in the proof of Lemma 12, we have that if $T \in G$, the G_{p_3} components of the challenge ciphertext are properly distributed as in a response from $O_{3.5}$, since the value of c modulo p_3 is not revealed by the challenge key-type response (it is hidden by the random term $Y_3^{r_1}$). Also as in the proof of Lemma 12, we have that the G_{p_2} components of the ciphertext-type responses are properly distributed. Thus, if $T \in G_{p_1 p 2}$, $\mathcal{O}$ has properly simulated the responses of $O_{3.6}$, and when $T \in G$, $\mathcal{O}$ has properly simulated the responses of $O_{3.6}$.

Hence, if a PPT attacker can distinguish any pair between G_{ESF-5} and G_{ESF-6} with non-negligible advantage, $\mathcal{O}$ can distinguish the corresponding pair between $O_{3.5}$ and $O_{3.6}$ with non-negligible advantage. It means $\mathcal{O}$ can use the output of $\mathcal{B}$ to achieve a non-negligible advantage against Assumption 4.

Thus, Under Assumptions 4, no PPT attacker can distinguish between $O_{3.5}$ and $O_{3.6}$ with non-negligible advantage. Thus, no PPT attacker can distinguish between $G_{ESF'-5}$ and $G_{ESF'-6}$ with non-negligible advantage.

Lemma 14 *Under Assumptions 4, no PPT attacker can distinguish between $O_{3.6}$ and $O_{7/2'}$ with non-negligible advantage. So no PPT attacker can distinguish between $F6_{i-1,2}$ and $F6_{i,1}$ with non-negligible advantage.*

Proof We assume $\mathcal{B}$ interacts with one of $O_{3.6}$ and $O_{7/2'}$. $\mathcal{O}$ receives $g, g_3, X_1 X_2, Y_2 Y_3, T$. $\mathcal{O}$ will simulate either $O_{3.6}$ and $O_{7/2'}$ with $\mathcal{B}$, depending on the value of T (which is either in G or $G_{p_1 p3}$). $\mathcal{O}$ picks values $a, b, c, d, a_0, b_0, c_0, d_0 \in Z_N$ uniformly at random and sets $u = g^a$, $h = g^b$, $v = g^c$, $w = g^d$, $u_0 = g^{a_0}$, $h_0 = g^{b_0}$, $v_0 = g^{c_0}$, $w_0 = g^{d_0}$. $\mathcal{B}$ initially obtains the group elements

$$\begin{gathered} g, u, h, v, w, X_1X_2, w^y(Y_2Y_3)^{y\psi_1}, g^y(Y_2Y_3)^y, v^y(Y_2Y_3)^{y\sigma_1} \\ u_0, h_0, v_0, w_0, w_0^{y'}(Y_2Y_3)^{y'\psi_2}, g^{y'}(Y_2Y_3)^{y'}, v_0^{y'}(Y_2Y_3)^{y'\sigma_2} \end{gathered} \tag{89}$$

from its oracle simulator who additionally chooses $\psi_1, \psi_2, \sigma_1, \sigma_2, y, y' \in Z_N$ randomly. These are properly distributed with g^s implicitly set to be X_1.

When $\mathcal{A}$ requests the challenge ciphertext for messages M_0, M_1, identity vector $(I_1^*, \cdots, I_l^*)$ and T^*, $\mathcal{B}$ makes a ciphertext-type query to the oracle for each I_i^* and T^*. When $\mathcal{B}$ makes a ciphertext-type query for some identity I_i^*, $\mathcal{O}$ responds by choosing a random $t_i' \in Z_N$ and returning $((X_1X_2)^d(X_1X_2)^{ct_i'}, (X_1X_2)^{t_i'(aI_i^*+b)}, (X_1X_2)^{t_i'})$ to $\mathcal{B}$. When $\mathcal{B}$ makes a ciphertext-type query for some time T^*, $\mathcal{O}$ responds by choosing a random $t_0 \in Z_N$ and returning $((X_1X_2)^{d_0}(X_1X_2)^{c_0t_0'}, (X_1X_2)^{t_0'(a_0T^*+b_0)}, (X_1X_2)^{t_0'})$ to $\mathcal{B}$. This sets $X_2^d = g_2^{\delta_1}$ and $X_2^{d_0} = g_2^{\delta_2}$, which is uniformly random because the value of d and d_0 modulo p_2 will not appear elsewhere. It implicitly sets $g^{t_i} = X_1^{t_i'}$. This is identically distributed to a response from O_6 and $O_{7/2'}$, with a', b' equal to a, b modulo p_2, and $\sigma_1 = c$ modulo p_2, $\sigma_2 = c_0$ modulo p_2. We note that this is in the only context in which the values of a, b modulo p_2 appear, so this is equivalent to choosing a', b' independently at random. Then $\mathcal{B}$ creats the ESF-1 ciphertexts successfully.

When $\mathcal{A}$ requests the secret key of an identity vector $ID|_j = (I_1, \cdots, I_j)$, $\mathcal{B}$ creats the ESF-4-SK key by the HIBE-type query responses from $\mathcal{O}$ who randomly chooses $y_1, \cdots, y_j$, $\lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_j, z, z' \in Z_n$ and generates a ESF-4-SK PSK_{HIBE,h_θ} for every θ

$$K_{i,0} = g^{\lambda_i}w^{y_i}, K_{i,1} = g^{y_i}, K_{i,2} = g^{r_i}, K_{i,3} = (u^{I_i}h)^{r_i}v^{y_i}, i \in \{1, \cdots, j-1\}$$

$$K_{j,0} = g^{\gamma_\theta - \sum_{i=1}^{j-1}\lambda_i}w^{y_j}(Y_2Y_3)^{y_j\psi_1}, K_{j,1} = g^{y_j}(Y_2Y_3)^{y_j},$$

$$K_{j,2} = v^{y_j}(u^{I_j}h)^{r_j}(Y_2Y_3)^z, K_{j,3} = g^{r_j}(Y_2Y_3)^{z'}$$

When $\mathcal{B}$ creats the IBE private key with the index pair (h, i_c) for some time $\mathcal{T}$ for the identity vector $(I_1, \cdots, I_{j-1})$ in the index h node, the update key with an index pair (h, i_c) is generated as follows:

1. $i_c < h_c$: It randomly chooses $y_0, \cdots, y_{j-1}, \lambda_1, \cdots, \lambda_{j-1}, r'_0, r_1, \cdots, r_{j-1}, z, z' \in Z_n$ and generates a semi-functional update key $TUK_{ID|_h,T,\theta h}$.

$$U_{0,0} = g^{\alpha-\gamma_{\theta_h}-\sum_{i=1}^{j-1}\lambda_i} w_0^{y_0}(Y_2Y_3)^{y_0\psi_2}, U_{0,1} = g^{y_0}(Y_2Y_3)^{y_0}, U_{0,2} = g^{r_0},$$
$$U_{0,3} = v_0^{y_0}(Y_2Y_3)^{y_0\sigma_2}(u_0^T h_0)^{r_0}$$
$$U_{i,0} = g^{\lambda_i} w^{y_i}, U_{i,1} = g^{y_i}, U_{i,2} = g^{r_i}, U_{i,3} = (u^{I_i}h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

2. $i_c = h_c$: $\mathcal{B}$ chooses random values $y_1, \cdots, y_{j-1}, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_{j-1} \in Z_n$. $\mathcal{B}$ forms the challenge key as:

$$U_{0,0} = g^{\alpha-\gamma_{\theta_h}-\sum_{i=1}^{j-1}\lambda_i} T_0, U_{0,1} = T_1, U_{0,2} = T_3, U_{0,3} = T_2$$
$$U_{i,0} = g^{\lambda_i} w^{y_i}, U_{i,1} = g^{y_i}, U_{i,2} = g^{r_i}, U_{i,3} = (u^{I_i}h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

where (T_0, T_1, T_2, T_3) is the challenge IBE key queried to $\mathcal{O}$ who chooses a random $y_0 \in Z_N$ and returns $(T_0, T_1, T_2, T_3) = (w_0^{y_0}(Y_2Y_3)^{\psi_2 y_0}, g^{y_0}(Y_2Y_3)^{y_0}, v_0^{y_0}(Y_2Y_3)^{\sigma_2 y_0} T^{a_0 T + b_0}, T)$ to $\mathcal{B}$.

3. $i_c > h_c$: It generates a ESF-4-UK as

$$U_{0,0} = g^{\alpha-\gamma_{\theta_h}-\sum_{i=1}^{j-1}\lambda_i}(w_0^{y_0}(Y_2Y_3)^{\psi_2 y_0}, U_{0,1} = g^{y_0}(Y_2Y_3)^{y_0}, U_{0,2} = g^{r_0}(Y_2Y_3)^{z},$$
$$U_{0,3} = v_0^{y_0}(u_0^T h_0)^{r_0}(Y_2Y_3)^{z'}$$
$$U_{i,0} = g^{\lambda_i} w^{y_i}, U_{i,1} = g^{y_i}, U_{i,2} = g^{r_i}, U_{i,3} = (u^{I_i}h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

where $z, z' \in Z_p$ are randomly chosen.

In the challenge IBE key, it implicitly sets g^r to be the G_{p_1} part of T. We note that a_0, b_0 modulo p_2, p_3 are uniformly random and do not appear elsewhere. If $T \in G_{p_1p3}$, then this matches the distribution of O_6, and so this will be a properly distributed normal key and $\mathcal{B}$ is playing Game $F6_{h_c-1,2}$. If $T \in G$, then this matches the distribution of $O_{7/2'}$ (note random G_{p_3} terms attached to the last two group elements) and then $\mathcal{B}$ is playing Game $F6_{h_c,1}$.

Hence, if a PPT attacker can distinguish between $F6_{h_c-1,2}$ and $F6_{h_c,1}$ with non-negligible advantage, $\mathcal{O}$ can distinguish between O_6 and $O_{7/2'}$ with non-negligible advantage. It means $\mathcal{O}$ can gain a non-negligible advantage against Assumption 4.

Thus, Under Assumptions 4, no PPT attacker can distinguish between O_6 and $O_{7/2'}$ with non-negligible advantage. Thus, no PPT attacker can distinguish between $F6_{h_c-1,2}$ and $F6_{h_c,1}$ with non-negligible advantage.

Lemma 15 *Under Assumptions 3, no PPT attacker can distinguish between $O_{7/2'}$ and $\widetilde{O}^*_{q_c}$ with non-negligible advantage. So no PPT attacker can distinguish between $F6_{i,1}$ and $F6_{i,2}$ with non-negligible advantage.*

Proof We assume $\mathcal{B}$ interacts with one of $O_{7/2'}$, and $\widetilde{O}^*_{q_c}$. $\mathcal{O}$ receives $g, g_2, X_1 X_3, T$. $\mathcal{O}$ will simulate either $O_{7/2'}$ or $\widetilde{O}^*_{q_c}$ with $\mathcal{B}$, depending on the value of T (which is either in G_{p_1} or G_{p_1p3}). $\mathcal{O}$ picks values $a, b, c, d, a_0, b_0, c_0, d_0 \in Z_N$ uniformly at random and sets $u = g^a$, $h = g^b$,

$v = g^c$, $w = g^d$, $u_0 = g^{a_0}$, $h_0 = g^{b_0}$, $v_0 = g^{c_0}$, $w_0 = g^{d_0}$. $\mathcal{B}$ initially obtains the group elements

$$\begin{gathered} g, u, h, v, w, g^s g_2^{\gamma}, (X_1X_3)^d g_2^{dy}, (X_1X_3) g_2^{y}, (X_1X_3)^c g_2^{y'}, \\ u_0, h_0, v_0, w_0, (X_1X_3)^{zd_0} g_2^{d_0 y'}, (X_1X_3)^z g_2^{y'}, (X_1X_3)^{zc_0} g_2^{y'} \end{gathered} \tag{90}$$

from its oracle simulator, and gives $\mathcal{A}$ the public parameters in Eq 83.

When $\mathcal{A}$ requests the challenge ciphertext for messages M_0, M_1, identity vector $(I_1^*, \cdots, I_l^*)$ and T^*, $\mathcal{B}$ makes a ciphertext-type query to the oracle for each I_i^* and T^*. When $\mathcal{B}$ makes a ciphertext-type query for some identity I^*, $\mathcal{O}$ responds by choosing a random $t \in Z_N$ and returning $(w^s g_2^{\delta_1} v^t g_2^{\sigma_1 t}, g^t g_2^t, (u^{I^*} h)^t g_2^{t(a'I^* + b')})$ to $\mathcal{B}$ as same as Eq 10. When $\mathcal{B}$ makes a ciphertext-type query for some time T^*, $\mathcal{O}$ responds by choosing a random $t_0 \in Z_N$ and returning $(w_0^s g_2^{\delta_2} v_0^{t_0} g_2^{\sigma_2 t_0}, g^{t_0} g_2^{t_0}, (u_0^{T^*} h_0)^{t_0} g_2^{t_0(a'T^* + b')})$ to $\mathcal{B}$. Then $\mathcal{B}$ creats the ESF-1 ciphertexts successfully.

When $\mathcal{A}$ requests the secret key of an identity vector $ID|_j = (I_1, \cdots, I_j)$, $\mathcal{B}$ creats the ESF-4-SK key by the HIBE-type query response from $\mathcal{O}$ and the secret key for $ID|_j$ in some node θ is

$$K_{i,0} = g^{\lambda_i} w^{y_i}, K_{i,1} = g^{y_i}, K_{i,2} = g^{r_i}, K_{i,3} = (u^{I_i} h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

$$K_{j,0} = g^{\gamma_\theta - \sum_{i=1}^{j-1} \lambda_i} (X_1X_3g_2)^{dy'_j}, K_{j,1} = (X_1X_3g_2)^{y'_j},$$

$$K_{j,2} = (X_1X_3)^{r'_j} g_2^{z'}, K_{j,3} = (X_1X_3g_2)^{cy'_j} (X_1X_3)^{r'_j(aI_j + b)} g_2^{z}$$

where $y_1, \cdots, y_{j-1}, y'_j, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_{j-1}, r'_j, z, z' \in Z_n$ are randomly chosen.

When $\mathcal{B}$ creats the IBE private key with the index pair (h, i_c) for some time $\mathcal{T}$ for the identity vector $(I_1, \cdots, I_{j-1})$ in the index h node, the update key with an index pair (h, i_c) is generated as follows:

1. $i_c < h_c$: It randomly chooses $y'_0, y_1, \cdots, y_{j-1}, \lambda_1, \cdots, \lambda_{j-1}, r_0, r_1, \cdots, r_{j-1}, z, z' \in Z_n$ and generates a semi-functional update key $TUK_{ID|_j T, \theta h}$

$$U_{0,0} = g^{\alpha - \gamma_{\theta_h} - \sum_{i=1}^{j-1} \lambda_i} (X_1X_3g_2)^{d_0 y'_0}, U_{0,1} = (X_1X_3g_2)^{y'_0}, U_{0,2} = g^{r_0},$$

$$U_{0,3} = (X_1X_3g_2)^{c_0 y'_0} (u_0^T h_0)^{r_0}$$

$$U_{i,0} = g^{\lambda_i} w^{y_i}, U_{i,1} = g^{y_i}, U_{i,2} = g^{r_i}, U_{i,3} = (u^{I_i} h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

2. $i_c = h_c$: $\mathcal{B}$ chooses random values $y_1, \cdots, y_{j-1}, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_{j-1} \in Z_n$. $\mathcal{B}$ forms the challenge key as:

$$U_{0,0} = g^{\alpha - \gamma_{\theta_h} - \sum_{i=1}^{j-1} \lambda_i} T_0, U_{0,1} = T_1, U_{0,2} = T_3, U_{0,3} = T_2$$

$$U_{i,0} = g^{\lambda_i} w^{y_i}, U_{i,1} = g^{y_i}, U_{i,2} = g^{r_i}, U_{i,3} = (u^{I_i} h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

where (T_0, T_1, T_2, T_3) is the challenge IBE key queried to $\mathcal{O}$ who chooses a random $y_0 \in Z_N$ and returns $(T_0, T_1, T_2, T_3) = ((X_1X_3g_2)^{d_0 y'_0}, (X_1X_3g_2)^{y'_0}, (X_1X_3g_2)^{c_0 y'_0} T^{a_0 T + b_0}, T)$. This implicitly sets g^r to be the G_{p_1} part of T.

3. $i_c > h_c$: It generates a ESF-4-UK as

$$U_{0,0} = g^{\alpha-\gamma_{\theta_h}-\sum_{i=1}^{j-1}\lambda_i}(X_1X_3g_2)^{d_0y'_0}, U_{0,1} = (X_1X_3g_2)^{y'_0},$$

$$U_{0,2} = (X_1X_3)^{r'_0}g_2^{z'}, U_{0,3} = (X_1X_3g_2)^{c_0y'_0}(X_1X_3)^{r'_0(a_0T+b_0)}g_2^{z}$$

$$U_{i,0} = g^{\lambda_i}w^{y_i}, U_{i,1} = g^{y_i}, U_{i,2} = g^{r_i}, U_{i,3} = (u^{I_i}h)^{r_i}v^{y_i}, i \in \{1,\cdots,j-1\}$$

where $z, z' \in Z_p$ are randomly chosen.

In the challenge IBE key, it implicitly sets g^{r_0} to be the G_{p_1} part of T. We note that a_0, b_0 modulo p_2, p_3 are uniformly random and do not appear elsewhere. If $T \in G_{p_1p3}$, then this matches the distribution of $O_{7/2'}$, and so this will be a properly distributed normal key and $\mathcal{B}$ is playing Game $F6_{h_c,1}$. If $T \in G_{p_1}$, then this matches the distribution of $\widetilde{O}^*_{q_c}$ and then $\mathcal{B}$ is playing Game $F6_{h_c,2}$.

Hence, if a PPT attacker can distinguish between $F6_{h_c,1}$ and $F6_{h_c,2}$ with non-negligible advantage, $\mathcal{O}$ can distinguish between $O_{7/2'}$ and $\widetilde{O}^*_{q_c}$ with non-negligible advantage. It means $\mathcal{O}$ can gain a non-negligible advantage against Assumption 3.

Thus, Under Assumptions 3, no PPT attacker can distinguish between $O_{7/2'}$ and $\widetilde{O}^*_{q_c}$ with non-negligible advantage. Thus, no PPT attacker can distinguish between $F6_{h_c,1}$ and $F6_{h_c,2}$ with non-negligible advantage.

Lemma 16 *Under Assumptions 3, no PPT attacker can distinguish between* $\widetilde{O}^*_k$ *and* $\widetilde{O}''_k$ *with non-negligible advantage. So no PPT attacker can distinguish between* $S'_{k,1}$ *and* $S'_{k,3}$ *with non-negligible advantage.*

Proof We assume $\mathcal{B}$ interacts with one of $\widetilde{O}''_k$, $\widetilde{O}^*_k$. $\mathcal{O}$ receives g, g_2, X_1 X_3, T. $\mathcal{O}$ will simulate either $\widetilde{O}''_k$ or $\widetilde{O}^*_k$ with $\mathcal{B}$, depending on the value of T (which is either in G_{p_1} or G_{p_1p3}). $\mathcal{B}$ initially obtains the group elements in Eq 82

$$g, u, h, v, w, g^s g_2^{\gamma}, (X_1X_3)^d g_2^{y_2d}, (X_1X_3)g_2^{y_2}, (X_1X_3)^c g_2^{cy_2}$$

$$u_0, h_0, v_0, w_0, (X_1X_3)^{zd_0}g_2^{y'_2d_0}, (X_1X_3)^z g_2^{y'_2}, (X_1X_3)^{zc_0}g_2^{c_0y'_2}$$

from its oracle simulator.

When $\mathcal{A}$ requests the secret key of an identity vector $ID|_j = (I_1, \cdots, I_j)$, $\mathcal{B}$ creats the ESF-4-SK key by the HIBE-type query response from $\mathcal{O}$ and the secret key for $ID|_j$ in some node θ is

$$K_{i,0} = g^{\lambda_i}w^{y_i}, K_{i,1} = g^{y_i}, K_{i,2} = g^{r_i}, K_{i,3} = (u^{I_i}h)^{r_i}v^{y_i}, i \in \{1,\cdots,j-1\}$$

$$K_{j,0} = g^{\gamma_\theta-\sum_{i=1}^{j-1}\lambda_i}(X_1X_3g_2)^{dy'_j}, K_{j,1} = (X_1X_3g_2)^{y'_j},$$

$$K_{j,2} = (X_1X_3)^{r'_j}g_2^{z'}, K_{j,3} = (X_1X_3g_2)^{cy'_j}(X_1X_3)^{r'_j(aI_j+b)}g_2^{z}$$

where $y_1, \cdots, y_{j-1}, y'_j, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_{j-1}, r'_j, z, z' \in Z_n$ are randomly chosen.

When $\mathcal{A}$ requests the update key of an identity vector $ID|_j = (I_1, \cdots, I_j)$ and the ime T, $\mathcal{B}$ creats the UK key by the IBE-type query response from $\mathcal{O}$ and the secret key for $ID|_j$ in some node θ is generated as follows: $\mathcal{O}$ randomly chooses $y'_0, y_1, \cdots, y_{j-1}, \lambda_1, \cdots, \lambda_{j-1}, r_0, r_1, \cdots, r_{j-1}$,

$z, z' \in Z_n$ and generates a semi-functional update key $TUK_{ID|_b T,\theta h}$

$$U_{0,0} = g^{\alpha-\gamma_{\theta_h}-\sum_{i=1}^{j-1}\lambda_i}(X_1X_3g_2)^{d_0y'_0}, U_{0,1} = (X_1X_3g_2)^{y'_0}, U_{0,2} = g^{r_0},$$

$$U_{0,3} = (X_1X_3g_2)^{c_0y'_0}(u_0^T h_0)^{r_0}$$

$$U_{i,0} = g^{\lambda_i}w^{y_i}, U_{i,1} = g^{y_i}, U_{i,2} = g^{r_i}, U_{i,3} = (u^{I_i}h)^{r_i}v^{y_i}, i \in \{1, \cdots, j-1\}$$

When $\mathcal{A}$ requests the challenge ciphertext for messages M_0, M_1, identity vector $(I_1^*, \cdots, I_l^*)$ and T^*, $\mathcal{B}$ makes a ciphertext-type query to the oracle for each I_i^* and T^*. In response to each query for I_i^* or T^*, $\mathcal{O}$ gets random $t_0, t_1, \ldots, t_l \in Z_p$, chooses $\beta \in \{0, 1\}$ and creats the ciphertext as

$$(C = M_\beta e(g^s g_2^\gamma, g)^\alpha, C_0 = g^s g_2^\gamma, \{C_{i,1}, C_{i,2}, C_{i,3}\}_{i=0}^l)$$

where the ciphertext-element-group ($C_{i,1}$, $C_{i,2}$, $C_{i,3}$) is defined as follows:

1. $i < k$: If $i = 0$, $\mathcal{O}$ responds with the ciphertext-element-group $(w_0^s g_2^{\delta_2} v_0^{t_0} g_2^{c_0 t_0}, (u_0^{T^*} h_0)^{t_0} g_2^{(a'T^*+b')t_0}, g^{t_0} g_2^{t_0})$, else the element group is $(w^s g_2^{\delta_1} v^{t_i} g_2^{ct_i}, (u^{I_i^*} h)^{t_i} g_2^{(a'I_i^*+b')t_i}, g^{t_i} g_2^{t_i})$. This is identically distributed to a response from O_2.
2. $i = k$: $\mathcal{O}$ choses $z \in Z_p$ randomly and responds with the ciphertext-element-group (T_1, T_3, T_2) = $(w^s g_2^{\delta_1} T^c \cdot g_2^{cz}, T^{t_i(aI_i^*+b)} g_2^{z(a'I_i^*+b')}, Tg_2^z)$ if $k > 0$. Else if $k = 0$, $\mathcal{O}$ responds with $(w_0^s g_2^{\delta_2} T^{c_0} \cdot g_2^{c_0 z}, T^{t_0(a_0T^*+b_0)} g_2^{z(a'T^*+b')}, Tg_2^z)$. We note that the G_{p_2} parts here are properly distributed, since $\sigma_1 = c$ modulo p_2 and $\sigma_2 = c_0$ modulo p_2.
3. $i > k$: The ciphertext-element-group is $(w^s g_2^{\delta_1} v^{t_i}, (u^{I_i^*} h)^{t_i}, g^{t_i})$. This is identically distributed to a response from O_1.

When $T \in G_{p_1}$ the values of a, b modulo p_3 only appear in the response to the challenge key-type query, which means that the G_{p_3} terms on the last two group elements there are uniformly random. Also, the response to the k^{th} ciphertext-type query is distributed exactly like a response from O_2. In this case, $\mathcal{O}$ has properly simulated the responses of O_k^* and this will be a properly distributed EST-2^k-CT and so $\mathcal{B}$ is playing Game $S'_{k,1}$.

When $T \in G_{p_1p3}$, we must argue that the values $aI + b$ and $aI_k^* + b$ appear uniformly random modulo p_3: this follows by pairwise independence of $aI + b$ as a function of I modulo p_3, since we have restricted the **Type-1** adversary to choose I and I_k^* so that $I \neq I_k^*$ modulo p_3 and a, b modulo p_3 only appear in these two values. Hence, $\mathcal{O}$ has produced a properly distributed EST-4^k-CT and $\mathcal{O}$ has properly simulated the response of O''_k in this case. So $\mathcal{B}$ is playing Game $S'_{k,3}$. We have thus shown that $\mathcal{O}$ can use the output of $\mathcal{B}$ to achieve non-negligible advantage against Assumption 3.

Hence, if a PPT attacker can distinguish any pair between $S'_{k,3}$ and $S'_{k,1}$ with non-negligible advantage, $\mathcal{O}$ can distinguish the corresponding pair between $\widetilde{O}''_k$ and $\widetilde{O}^*_k$ with non-negligible advantage. It means $\mathcal{O}$ can use the output of $\mathcal{B}$ to achieve a non-negligible advantage against Assumption 3.

Thus, Under Assumptions 3, no PPT attacker can distinguish between $\widetilde{O}''_k$ and $\widetilde{O}^*_k$ with non-negligible advantage. Thus, no PPT attacker can distinguish between $S'_{k,3}$ and $S'_{k,1}$ with non-negligible advantage.

Lemma 17 *Under Assumptions 4, no PPT attacker can distinguish between* $\widetilde{O}''_k$ *and* $\widetilde{O}'_k$ *with non-negligible advantage. So no PPT attacker can distinguish between* $S'_{k,3}$ *and* $S'_{k,2}$ *with non-negligible advantage.*

Proof We assume $\mathcal{B}$ interacts with one of $\widetilde{O}'_k$, $\widetilde{O}''_k$. $\mathcal{O}$ receives g, g_3, X_1 X_2, Y_2 Y_3, T. $\mathcal{O}$ will simulate either $\widetilde{O}'_k$ or $\widetilde{O}''_k$ with $\mathcal{B}$, depending on the value of T (which is either in G or $G_{p_1 p3}$). $\mathcal{O}$ picks values a, b, c, d, a_0, b_0, c_0, $d_0 \in Z_N$ uniformly at random and sets $u = g^a$, $h = g^b$, $v = g^c$, $w = g^d$, $u_0 = g^{a_0}$, $h_0 = g^{b_0}$, $v_0 = g^{c_0}$, $w_0 = g^{d_0}$. $\mathcal{B}$ initially obtains the group elements in Eq 85

$$g, u, v, w, X_1X_2, w^y(Y_2Y_3)^{\psi}, g^y(Y_2Y_3), v^y(Y_2Y_3)^c,$$

$$u_0, v_0, w_0, w_0^{y_0}(Y_2Y_3)^{z\psi}, g^{y_0}(Y_2Y_3)^z, v_0^{y_0}(Y_2Y_3)^{zc_0}$$

from its oracle simulator where z, y_0, y, $\psi \in Z_p$ are randomly chosen.

When $\mathcal{A}$ makes a secret key query for the identity $ID|_j = (I_1, \cdots, I_j)$, then $\mathcal{B}$ makes its challenge HIBE-key-type query for I_j, $\mathcal{O}$ chooses r, y', z, $z' \in Z_n$ randomly and returns the group elements

$$(w^{y'}(Y_2Y_3)^{y'\psi_1}, g^{y'}(Y_2Y_3)^{y'}, g^r(Y_2Y_3)^z, v^{y'}(u^Ih)^r(Y_2Y_3)^{z'})$$

to $\mathcal{B}$. And then $\mathcal{B}$ creats the ESF-4 secret key by using the group elements.

Upon receiving a challenge IBE-key-type query for $T \in Z_n$, $\mathcal{O}$ chooses r_0, $y_0 \in Z_n$ randomly and returns the group elements

$$(w_0^{y_0}(Y_2Y_3)^{y_0\psi_2}, g^{y_0}(Y_2Y_3)^{y_0}, g^{r_0}, v_0^{y_0}(Y_2Y_3)^{y_0\sigma_2}(u_0^Th_0)^{r_0})$$

to $\mathcal{B}$. And then $\mathcal{B}$ creats the semi-functional update key by using the group elements.

When $\mathcal{A}$ requests the challenge ciphertext for messages M_0, M_1, identity vector $(I_1^*, \cdots, I_l^*)$ and T^*, $\mathcal{B}$ makes a ciphertext-type query to the oracle for each I_i^* and T^*. In response to each query for I_i^* or T^*, $\mathcal{O}$ gets random $t'_0, t'_1, \cdots, t'_k, t_{k+1} \cdots, t_l \in Z_p$, chooses $\beta \in \{0, 1\}$ and creats the ciphertext as

$$(C = M_\beta e(X_1X_2, g)^\alpha, C_0 = X_1X_2, \{C_{i,1}, C_{i,2}, C_{i,3}\}_{i=0}^l)$$

where the ciphertext-element-group $(C_{i,1}, C_{i,2}, C_{i,3})$ is defined as follows:

1. $i < k$: If $i = 0$, the ciphertext-element-group is

$$((X_1X_2)^{d_0}(X_1X_2)^{c_0t'_0}, (X_1X_2)^{t'_0(a_0T^*+b_0)}, (X_1X_2)^{t'_0})$$

else the ciphertext-element-group is

$$((X_1X_2)^{d}(X_1X_2)^{ct'_i}, (X_1X_2)^{t'_i(aI_i^*+b)}, (X_1X_2)^{t'_i})$$

This sets $X_2^d = g_2^{\delta_1}$ and $X_2^{d_0} = g_2^{\delta_2}$, which is uniformly random because the value of d and d_0 modulo p_2 will not appear elsewhere. It implicitly sets $g^{t_i} = X_1^{t'_i}$. This is identically distributed to a response from O_2, with a', b' equal to a, b modulo p_2, and $\sigma_1 = c$ modulo p_2, $\sigma_2 = c_0$ modulo p_2. We note that this is in the only context in which the values of a, b modulo p_2 appear, so this is equivalent to choosing a', b' independently at random.

2. $i = k$: If $k = 0$, the ciphertext-element-group is $(T_1, T_3, T_2) = ((X_1X_2)^{d_0}T^{c_0}, T^{(a_0T^*+b_0)}, T)$; If $k > 0$, the ciphertext-element-group is $(T_1, T_3, T_2) = ((X_1X_2)^{d}T^{c}, T^{(aI_i^*+b)}, T)$;

3. $i > k$: The ciphertext-element-group is $((X_1X_2)^d v^{t_i}, (u^{I_i^*}h)^{t_i}, g^{t_i})$.

If $T \in G_{p_1p3}$, then the response for ciphertext-type query i is identically distributed to a response from $\widetilde{O}'_k$.

In this case, $\mathcal{O}$ has produced a properly distributed EST-3^k-CT and $\mathcal{B}$ is playing Game $S'_{k,2}$.

If $T \in G$, then this response additionally has terms in G_{p_2} which are appropriately distributed with $c = \sigma_1$, $a = a'$, $b = b'$ modulo p_2 or $c_0 = \sigma_2.a_0 = a'$, $b_0 = b'$ modulo p_2. Thus, the response is identically distributed to a response from $\widetilde{O}''_k$. In this case, $\mathcal{O}$ has produced a properly distributed EST-4^k-CT and $\mathcal{B}$ is playing Game $S'_{k,3}$.

Hence, if a PPT attacker can distinguish any pair between $S'_{k,2}$ and $S'_{k,3}$ with non-negligible advantage, $\mathcal{O}$ can distinguish the corresponding pair between $\widetilde{O}'_k$ and $\widetilde{O}''_k$ with non-negligible advantage. It means $\mathcal{O}$ can use the output of $\mathcal{B}$ to achieve a non-negligible advantage against Assumption 4.

Thus, Under Assumptions 4, no PPT attacker can distinguish between $\widetilde{O}'_k$ and $\widetilde{O}''_k$ with non-negligible advantage. Thus, no PPT attacker can distinguish between $S'_{k,2}$ and $S'_{k,3}$ with non-negligible advantage.

Lemma 18 *Under Assumptions 3, no PPT attacker can distinguish between* $\widetilde{O}'_k$ *and* $\widetilde{O}^*_{k-1}$ *with non-negligible advantage. So no PPT attacker can distinguish between* $S'_{k,2}$ *and* $S'_{k-1,1}$ *with non-negligible advantage.*

Proof We assume $\mathcal{B}$ interacts with one of $\widetilde{O}^*_{k-1}$, $\widetilde{O}'_k$. $\mathcal{O}$ receives $g, g_2, X_1 X_3, Y_1 Y_3, T$. $\mathcal{O}$ will simulate either $\widetilde{O}^*_{k-1}$ or $\widetilde{O}'_k$ with $\mathcal{B}$, depending on the value of T (which is either in G_{p_1} or G_{p_1p3}). $\mathcal{B}$ initially obtains the group elements in Eq 82

$$g, u, h, v, w, g^s g_2^{\gamma}, (X_1X_3)^d g_2^{y_2 d}, (X_1X_3) g_2^{y_2}, (X_1X_3)^c g_2^{cy_2}$$

$$u_0, h_0, v_0, w_0, (X_1X_3)^{zd_0} g_2^{y'_2 d_0}, (X_1X_3)^z g_2^{y'_2}, (X_1X_3)^{zc_0} g_2^{c_0 y'_2}$$

from its oracle simulator. It chooses $\alpha \in Z_n$ randomly, and gives $\mathcal{A}$ the public parameters in Eq 83. B can responds by using the normal update key generation and the normal decryption key derivation algorithm, since it knows α.

When $\mathcal{A}$ requests the secret key of an identity vector $ID|_j = (I_1, \cdots, I_j)$, $\mathcal{B}$ creats the ESF-4-SK key by the HIBE-type query response from $\mathcal{O}$ and the secret key for $ID|_j$ in some node θ is

$$K_{i,0} = g^{\lambda_i} w^{y_i}, K_{i,1} = g^{y_i}, K_{i,2} = g^{r_i}, K_{i,3} = (u^{I_i} h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

$$K_{j,0} = g^{\gamma_\theta - \sum_{i=1}^{j-1} \lambda_i} (X_1X_3g_2)^{dy'_j}, K_{j,1} = (X_1X_3g_2)^{y'_j},$$

$$K_{j,2} = (X_1X_3)^{r'_j} g_2^{z'}, K_{j,3} = (X_1X_3g_2)^{cy'_j} (X_1X_3)^{r'_j(aI_j+b)} g_2^{z}$$

where $y_1, \cdots, y_{j-1}, y'_j, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_{j-1}, r'_j, z, z' \in Z_n$ are randomly chosen.

When $\mathcal{A}$ requests the update key of an identity vector $ID|_j = (I_1, \cdots, I_j)$ and the ime T, $\mathcal{B}$ creats the UK key by the IBE-type query response from $\mathcal{O}$ and the secret key for $ID|_j$ in some node θ is generated as follows: $\mathcal{O}$ randomly chooses $y'_0, y_1, \cdots, y_{j-1}, \lambda_1, \cdots, \lambda_{j-1}, r_0, r_1, \cdots, r_{j-1}$,

$z, z' \in Z_n$ and generates a semi-functional update key $TUK_{ID|_b T,\theta h}$

$$U_{0,0} = g^{\alpha-\gamma_{\theta_h}-\sum_{i=1}^{j-1}\lambda_i}(X_1X_3g_2)^{d_0y'_0}, U_{0,1} = (X_1X_3g_2)^{y'_0}, U_{0,2} = g^{r_0},$$

$$U_{0,3} = (X_1X_3g_2)^{c_0y'_0}(u_0^T h_0)^{r_0}$$

$$U_{i,0} = g^{\lambda_i}w^{y_i}, U_{i,1} = g^{y_i}, U_{i,2} = g^{r_i}, U_{i,3} = (u^{I_i}h)^{r_i}v^{y_i}, i \in \{1, \cdots, j-1\}$$

When $\mathcal{A}$ requests the challenge ciphertext for messages M_0, M_1, identity vector $(I_1^*, \cdots, I_l^*)$ and T^*, $\mathcal{B}$ makes a ciphertext-type query to the oracle for each I_i^* and T^*. In response to each query for I_i^* or T^*, $\mathcal{O}$ gets random $t_0, t_1, \ldots, t_l \in Z_p$, chooses $\beta \in \{0, 1\}$ and creats the ciphertext as

$$(C = M_\beta e(g^s g_2^\gamma, g)^\alpha, C_0 = g^s g_2^\gamma, \{C_{i,1}, C_{i,2}, C_{i,3}\}_{i=0}^l)$$

where the ciphertext-element-group $(C_{i,1}, C_{i,2}, C_{i,3})$ is defined as follows:

1. $i < k$: If $i = 0$, $\mathcal{O}$ and responds with the ciphertext-element-group $(w^s g_2^{\delta_2} v_0^{t_0} g_2^{c_0 t_0}, (u_0^{T^*} h_0)^{t_0} g_2^{(a'T^*+b')t_0}, g^{t_0} g_2^{t_0})$, else the element group is $(w^s g_2^{\delta_1} v^{t_i} g_2^{ct_i}, (u^{I_i^*} h)^{t_i} g_2^{(a'I_i^*+b')t_i}, g^{t_i} g_2^{t_i})$;
2. $i = k$: The ciphertext-element-group is $(T_1, T_3, T_2) = (w^s g_2^{\delta_1} T^c, T^{t_i(aI_i^*+b)}, T)$;
3. $i > k$: The ciphertext-element-group is $(w^s g_2^{\delta_1} v^{t_i}, (u^{I_i^*} h)^{t_i}, g^{t_i})$.

We must now argue that the challenge key-type query and the k^{th} ciphertext-type query responses are properly distributed. If $T \in G_{p_1}$, then the response to the k ciphertext type query is identically distributed to a response from O_1, and the values a, b modulo p_3 only appear in the response to the challenge key-type query, hence the G_{p_3} parts on the last two group elements here appear random in G_{p_3}. This will be a properly distributed EST-2^{k-1}-CT which means that the responses of $\mathcal{O}$ properly simulate the responses of $\widetilde{O}_{k-1}^*$ and $\mathcal{B}$ is playing Game $'S_{k-1,1}$.

If $T \in G_{p_1 p3}$, then we must argue that $aI + b$ and $aI_k^* + b$ both appear to be uniformly random modulo p_3: this follows from pairwise independence of the function $aI + b$ modulo p_3, since we have restricted the **Type-1** adversary to choose I and I_k^* so that $I \neq I_k^*$ modulo p_3. This means that the G_{p_3} components on the last two group elements of the challenge key-type query response and on the k ciphertext-type query response are uniformly random in the attacker's view. In this case, $\mathcal{O}$ has produced a properly distributed EST-3^k-CT which means that $\mathcal{O}$ has properly simulated the responses of $\widetilde{O}'_k$ and $\mathcal{B}$ is playing Game $S'_{k,2}$.

Hence, if a PPT attacker can distinguish any pair between $S'_{k-1,1}$ and $S'_{k,2}$ with non-negligible advantage, $\mathcal{O}$ can distinguish the corresponding pair between $\widetilde{O}_{k-1}^*$ and $\widetilde{O}'_k$ with non-negligible advantage. It means $\mathcal{O}$ can use the output of $\mathcal{B}$ to achieve a non-negligible advantage against Assumption 3.

Thus, Under Assumptions 3, no PPT attacker can distinguish between $\widetilde{O}_{k-1}^*$ and $\widetilde{O}'_k$ with non-negligible advantage. Thus, no PPT attacker can distinguish between $S'_{k-1,1}$ and $S'_{k,2}$ with non-negligible advantage.

Lemma 19 *Under Assumptions 4, no PPT attacker can distinguish between* $\widetilde{O}_0^*$ *and* $O_{7/2}$ *with non-negligible advantage. So no PPT attacker can distinguish between* $I_{h_c-1,2}$ *and* $I_{h_c,1}$ *with non-negligible advantage.*

Proof We assume $\mathcal{B}$ interacts with one of $\widetilde{O}_0^*$, $O_{7/2}$. $\mathcal{O}$ receives g, g_3, X_1 X_2, Y_2 Y_3, T. $\mathcal{O}$ will simulate either $\widetilde{O}_0^*$ or $O_{7/2}$ with $\mathcal{B}$, depending on the value of T (which is either in G or G_{p_1p3}). $\mathcal{O}$ picks values a, b, c, d, a_0, b_0, c_0, $d_0 \in Z_N$ uniformly at random and sets $u = g^a$, $h = g^b$, $v = g^c$, $w = g^d$, $u_0 = g^{a_0}$, $h_0 = g^{b_0}$, $v_0 = g^{c_0}$, $w_0 = g^{d_0}$. $\mathcal{B}$ initially obtains the group elements in Eq 89

$$g, u, h, v, w, X_1X_2, w^y(Y_2Y_3)^{y\psi_1}, g^y(Y_2Y_3)^y, v^y(Y_2Y_3)^{y\sigma_1}$$

$$u_0, h_0, v_0, w_0, w_0^{y'}(Y_2Y_3)^{y'\psi_2}, g^{y'}(Y_2Y_3)^{y'}, v_0^{y'}(Y_2Y_3)^{y'\sigma_2}$$

from its oracle simulator who additionally chooses ψ_1, ψ_2, σ_1, σ_2, y, $y' \in Z_N$ randomly. These are properly distributed with g^s implicitly set to be X_1.

When $\mathcal{A}$ requests the challenge ciphertext for messages M_0, M_1, identity vector $(I_1^*, \cdots, I_l^*)$ and T^*, $\mathcal{B}$ makes a ciphertext-type query to the oracle for each I_i^* and T^*. When $\mathcal{B}$ makes a ciphertext-type query for some identity I_i^*, $\mathcal{O}$ responds by choosing a random $t_i' \in Z_N$ and returning $((X_1X_2)^d v^{t_i}, (u^{I_i^*}h)^{t_i}, g^{t_i})$ to $\mathcal{B}$. When $\mathcal{B}$ makes a ciphertext-type query for some time T^*, $\mathcal{O}$ responds by choosing a random $t_0 \in Z_N$ and returning $((X_1X_2)^{d_0} v_0^{t_0}, (u_0^{T^*}h)^{t_0}, g^{t_0})$ to $\mathcal{B}$. (Note that this implicitly sets $g_2^{\delta_1} = X_2^d$ and $g_2^{\delta_2} = X_2^{d_0}$, which is uniformly random because the value of d and d_0 modulo p_2 does not occur elsewhere.) Then $\mathcal{B}$ creats the semi-functional ciphertexts successfully.

Upon receiving a challenge IBE-key-type query for $T \in Z_n$, $\mathcal{O}$ chooses r_0, $y_0 \in Z_n$ randomly and returns the group elements

$$(w_0^{y_0}(Y_2Y_3)^{y_0\psi_2}, g^{y_0}(Y_2Y_3)^{y_0}, g^{r_0}, v_0^{y_0}(Y_2Y_3)^{y_0\sigma_2}(u_0^Th_0)^{r_0})$$

to $\mathcal{B}$. And then $\mathcal{B}$ creats the semi-functional update key by using the group elements.

When $\mathcal{B}$ creats the HIBE private key with the index pair (h, i_c) for the identity vector $(I_1, \cdots, I_{j-1})$ in the index h node, the secret key with an index pair (h, i_c) is generated as follows:

1. $i_c < h_c$: It randomly chooses $y_1, \cdots, y_j, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_j, z, z' \in Z_n$ and generates a semi-functional secret key $TUK_{ID|_hT,\theta h}$.

$$K_{i,0} = g^{\lambda_i}w^{y_i}, K_{i,1} = g^{y_i}, K_{i,2} = g^{r_i}, K_{i,3} = (u^{I_i}h)^{r_i}v^{y_i}, i \in \{1, \cdots, j-1\}$$

$$K_{j,0} = g^{\gamma\theta_h - \sum_{i=1}^{j-1}\lambda_i}w^{y_j}(Y_2Y_3)^{y_j\psi_1}, K_{j,1} = g^{y_j}(Y_2Y_3)^{y_j}, K_{j,2} = g^{r_j},$$

$$K_{j,3} = v^{y_j}(Y_2Y_3)^{y_j\sigma_1}(u^{I_j}h)^{r_j}$$

2. $i_c = h_c$: $\mathcal{B}$ chooses random values $y_1, \cdots, y_{j-1}, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_{j-1} \in Z_n$. $\mathcal{B}$ forms the challenge key as:

$$K_{i,0} = g^{\lambda_i}w^{y_i}, K_{i,1} = g^{y_i}, K_{i,2} = g^{r_i}, K_{i,3} = (u^{I_i}h)^{r_i}v^{y_i}, i \in \{1, \cdots, j-1\}$$

$$K_{j,0} = g^{\gamma\theta_h - \sum_{i=1}^{j-1}\lambda_i}T_0, K_{j,1} = T_1, K_{j,2} = T_3, K_{j,3} = T_2$$

where (T_0, T_1, T_2, T_3) is the challenge HIBE key queried to $\mathcal{O}$ who chooses a random $y_j \in Z_N$ and returns $(T_0, T_1, T_2, T_3) = (w^{y_j}(Y_2Y_3)^{\psi_1 y_j}, g^{y_j}(Y_2Y_3)^{y_j}, v^{y_j}(Y_2Y_3)^{\sigma_1 y_j}T^{aI_j+b}, T)$ to $\mathcal{B}$.

3. $i_c > h_c$: It generates a ESF-4-SK as

$$K_{i,0} = g^{\lambda_i} w^{y_i}, K_{i,1} = g^{y_i}, K_{i,2} = g^{r_i}, K_{i,3} = (u^{I_i} h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

$$K_{j,0} = g^{\gamma_{\theta_h} - \sum_{i=1}^{j-1} \lambda_i} (w^{y_j} (Y_2 Y_3)^{\psi_1 y_j}, K_{j,1} = g^{y_j} (Y_2 Y_3)^{y_j}, K_{j,2} = g^{r_j} (Y_2 Y_3)^{z},$$

$$K_{j,3} = v^{y_j} (u^{I_j} h)^{r_j} (Y_2 Y_3)^{z'}$$

where $z, z' \in Z_p$ are randomly chosen.

In the challenge HIBE key, it implicitly sets g^{r_j} to be the G_{p_1} part of T. We note that a, b modulo p_2, p_3 are uniformly random and do not appear elsewhere. If $T \in G_{p_1 p 3}$, then this matches the distribution of $\widetilde{O}_0^*$, and so this will be a properly distributed ESF-4-SK key and $\mathcal{B}$ is playing Game $I_{h_c-1,2}$. If $T \in G$, then this matches the distribution of $O_{7/2}$ (note random G_{p_3} terms attached to the last two group elements) and then $\mathcal{B}$ is playing Game $I_{h_c,1}$.

Hence, if a PPT attacker can distinguish between $I_{h_c-1,2}$ and $I_{h_c,1}$ with non-negligible advantage, $\mathcal{O}$ can distinguish between $\widetilde{O}_0^*$ and $O_{7/2}$ with non-negligible advantage. It means $\mathcal{O}$ can gain a non-negligible advantage against Assumption 4.

Thus, Under Assumptions 4, no PPT attacker can distinguish between $\widetilde{O}_0^*$ and $O_{7/2}$ with non-negligible advantage. Thus, no PPT attacker can distinguish between $I_{h_c-1,2}$ and $I_{h_c,1}$ with non-negligible advantage.

Lemma 20 *Under Assumptions 3, no PPT attacker can distinguish between $O_{7/2}$ and O_4 with non-negligible advantage. So no PPT attacker can distinguish between $I_{h_c,1}$ and $I_{h_c,2}$ with non-negligible advantage.*

Proof We assume $\mathcal{B}$ interacts with one of $O_{7/2}$, and O_4. $\mathcal{O}$ receives g, g_2, X_1 X_3, T. $\mathcal{O}$ will simulate either $O_{7/2}$ or O_4 with $\mathcal{B}$, depending on the value of T (which is either in G_{p_1} or $G_{p_1 p3}$). $\mathcal{O}$ picks values a, b, c, d, a_0, b_0, c_0, $d_0 \in Z_N$ uniformly at random and sets $u = g^a$, $h = g^b$, $v = g^c$, $w = g^d$, $u_0 = g^{a_0}$, $h_0 = g^{b_0}$, $v_0 = g^{c_0}$, $w_0 = g^{d_0}$. $\mathcal{B}$ initially obtains the group elements in Eq 90

$$g, u, h, v, w, g^s g_2^{\gamma}, (X_1 X_3)^d g_2^{dy}, (X_1 X_3) g_2^{y}, (X_1 X_3)^c g_2^{y'},$$

$$u_0, h_0, v_0, w_0, (X_1 X_3)^{z d_0} g_2^{d_0 y'}, (X_1 X_3)^{z} g_2^{y'}, (X_1 X_3)^{z c_0} g_2^{y'}$$

from its oracle simulator, and gives $\mathcal{A}$ the public parameters in Eq 83.

When $\mathcal{A}$ requests the challenge ciphertext for messages M_0, M_1, identity vector $(I_1^*, \cdots, I_l^*)$ and T^*, $\mathcal{B}$ makes a ciphertext-type query to the oracle for each I_i^* and T^*. When $\mathcal{B}$ makes a ciphertext-type query for some identity I^*, $\mathcal{O}$ responds by choosing a random $t \in Z_N$ and returning $(w^s g_2^{\delta_1} v^t, g^t, (u^{I^*} h)^t)$ to $\mathcal{B}$. When $\mathcal{B}$ makes a ciphertext-type query for some time T^*, $\mathcal{O}$ responds by choosing a random $t_0 \in Z_N$ and returning $(w_0^s g_2^{\delta_2} v_0^{t_0}, g^{t_0}, (u_0^{T^*} h_0)^{t_0})$ to $\mathcal{B}$. Then $\mathcal{B}$ creats the semi-functional ciphertexts successfully.

Upon receiving a challenge IBE-key-type query for $T \in Z_n$, $\mathcal{O}$ chooses $r_0, y_0' \in Z_n$ randomly and returns the group elements

$$(X_1 X_3 g_2)^{d_0 y_0'}, (X_1 X_3 g_2)^{y_0'}, g^{r_0}, (X_1 X_3 g_2)^{c_0 y_0'} (u_0^T h_0)^{r_0}$$

to $\mathcal{B}$. And then $\mathcal{B}$ creats the semi-functional update key by using the group elements.

When $\mathcal{B}$ creats the HIBE private key with the index pair (h, i_c) for the identity vector $(I_1, \cdots, I_{j-1})$ in the index h node, the secret key with an index pair (h, i_c) is generated as follows:

1. $i_c < h_c$: It randomly chooses $y'_j, y_1, \cdots, y_{j-1}, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_j, z, z' \in Z_n$ and generates a semi-functional secret key $PSK_{ID|_j,\theta h}$

$$K_{i,0} = g^{\lambda_i} w^{y_i}, K_{i,1} = g^{y_i}, K_{i,2} = g^{r_i}, K_{i,3} = (u^{I_i} h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

$$K_{j,0} = g^{\gamma_{\theta_h} - \sum_{i=1}^{j-1} \lambda_i} (X_1 X_3 g_2)^{dy'_j}, K_{j,1} = (X_1 X_3 g_2)^{y'_j}, K_{j,2} = g^{r_j},$$

$$K_{j,3} = (X_1 X_3 g_2)^{cy'_j} (u^{I_j} h)^{r_j}$$

2. $i_c = h_c$: $\mathcal{B}$ chooses random values $y_1, \cdots, y_{j-1}, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_{j-1} \in Z_n$. $\mathcal{B}$ forms the challenge key as:

$$K_{i,0} = g^{\lambda_i} w^{y_i}, K_{i,1} = g^{y_i}, K_{i,2} = g^{r_i}, K_{i,3} = (u^{I_i} h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

$$K_{j,0} = g^{\gamma_{\theta_h} - \sum_{i=1}^{j-1} \lambda_i} T_0, K_{j,1} = T_1, K_{j,2} = T_3, K_{j,3} = T_2$$

where (T_0, T_1, T_2, T_3) is the challenge HIBE key queried to $\mathcal{O}$ who chooses a random $y'_j \in Z_N$ and returns $(T_0, T_1, T_2, T_3) = ((X_1 X_3 g_2)^{dy'_j}, (X_1 X_3 g_2)^{y'_j}, (X_1 X_3 g_2)^{cy'_j} T^{aI_j + b}, T)$. This implicitly sets g^{r_j} to be the G_{p_1} part of T.

3. $i_c > h_c$: It generates a ESF-4-SK as

$$K_{i,0} = g^{\lambda_i} w^{y_i}, K_{i,1} = g^{y_i}, K_{i,2} = g^{r_i}, K_{i,3} = (u^{I_i} h)^{r_i} v^{y_i}, i \in \{1, \cdots, j-1\}$$

$$K_{j,0} = g^{\gamma_{\theta_h} - \sum_{i=1}^{j-1} \lambda_i} (X_1 X_3 g_2)^{dy'_j}, K_{j,1} = (X_1 X_3 g_2)^{y'_j},$$

$$K_{j,2} = (X_1 X_3)^{r'_j} g_2^{z'}, K_{j,3} = (X_1 X_3 g_2)^{cy'_j} (X_1 X_3)^{r'_j (aI_j + b)} g_2^{z}$$

where $z, z' \in Z_p$ are randomly chosen.

In the challenge HIBE key, it implicitly sets g^{r_j} to be the G_{p_1} part of T. We note that a, b modulo p_2, p_3 are uniformly random and do not appear elsewhere. If $T \in G_{p_1 p_3}$, then this matches the distribution of $O_{7/2}$, and so this will be a properly distributed normal key and $\mathcal{B}$ is playing Game $I_{h_c 1}$. If $T \in G_{p_1}$, then this matches the distribution of O_4 and then $\mathcal{B}$ is playing Game $I_{h_c 2}$.

Hence, if a PPT attacker can distinguish between $I_{h_c 1}$ and $I_{h_c 2}$ with non-negligible advantage, $\mathcal{O}$ can distinguish between $O_{7/2}$ and O_4 with non-negligible advantage. It means $\mathcal{O}$ can gain a non-negligible advantage against Assumption 3.

Thus, Under Assumptions 3, no PPT attacker can distinguish between $O_{7/2}$ and O_4 with non-negligible advantage. Thus, no PPT attacker can distinguish between $I_{h_c 1}$ and $I_{h_c 2}$ with non-negligible advantage.

Acknowledgments

This research is supported by the project of the National Basic Research and Development Program of China (973 Program) No. 2012CB315906 and the National Key Research and Development Program 2017YFB0802301.

Author Contributions

Conceptualization: Qianqian Xing, Baosheng Wang.

Formal analysis: Qianqian Xing, Xiaofeng Wang, Jing Tao.

Funding acquisition: Baosheng Wang, Xiaofeng Wang.

Investigation: Qianqian Xing, Xiaofeng Wang, Jing Tao.

Methodology: Qianqian Xing, Xiaofeng Wang.

Project administration: Baosheng Wang, Xiaofeng Wang.

Resources: Qianqian Xing, Baosheng Wang.

Supervision: Baosheng Wang.

Validation: Qianqian Xing.

Writing – original draft: Qianqian Xing.

Writing – review & editing: Qianqian Xing, Xiaofeng Wang, Jing Tao.

References

1. Seo JH, Emura K. Efficient Delegation of Key Generation and Revocation Functionalities in Identity-Based Encryption. In: CT-RSA. vol. 7779. Springer; 2013. p. 343–358.
2. Horwitz J, Lynn B. Toward hierarchical identity-based encryption. In: Advances in Cryptology-EUROCRYPT 2002. Springer; 2002. p. 466–481.
3. Seo JH, Emura K. Revocable hierarchical identity-based encryption: history-free update, security against insiders, and short ciphertexts. In: Cryptographers Track at the RSA Conference. Springer; 2015. p. 106–123.
4. Tsai TT, Tseng YM, Wu TY. RHIBE: constructing revocable hierarchical ID-based encryption from HIBE. Informatica. 2014; 25(2):299–326. https://doi.org/10.15388/Informatica.2014.16
5. Lee K. Revocable Hierarchical Identity-Based Encryption with Adaptive Security. IACR Cryptology ePrint Archive. 2016; 2016:749.
6. Seo JH, Emura K. Adaptive-ID secure revocable hierarchical identity-based encryption. In: International Workshop on Security. Springer; 2015. p. 21–38.
7. Ryu G, Lee K, Park S, Lee DH. Unbounded hierarchical identity-based encryption with efficient revocation. In: International Workshop on Information Security Applications. Springer; 2015. p. 122–133.
8. Rouselakis Y, Waters B. Practical constructions and new proof methods for large universe attribute-based encryption. In: Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security. ACM; 2013. p. 463–474.
9. Xing Q, Wang B, Wang X, Chen P, Yu B, Tang Y, et al. Unbounded Revocable Hierarchical Identity-Based Encryption with Adaptive-ID Security. In: High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS), 2016 IEEE 18th International Conference on. IEEE; 2016. p. 430–437.
10. Waters B, et al. Dual System Encryption: Realizing Fully Secure IBE and HIBE under Simple Assumptions. In: Crypto. vol. 5677. Springer; 2009. p. 619–636.
11. Lewko AB, Waters B. New Techniques for Dual System Encryption and Fully Secure HIBE with Short Ciphertexts. In: TCC. vol. 5978. Springer; 2010. p. 455–479.
12. Lewko AB, Waters B. Unbounded HIBE and Attribute-Based Encryption. In: Eurocrypt. vol. 6632. Springer; 2011. p. 547–567.
13. Lee K, Park S. Revocable Hierarchical Identity-Based Encryption with Shorter Private Keys and Update Keys. IACR Cryptology ePrint Archive. 2016; 2016:460.
14. Seo JH, Emura K. Revocable identity-based cryptosystem revisited: Security models and constructions. IEEE Transactions on Information Forensics and Security. 2014; 9(7):1193–1205. https://doi.org/10.1109/TIFS.2014.2327758
15. Naor D, Naor M, Lotspiech J. Revocation and tracing schemes for stateless receivers. In: Advances in Cryptology-CRYPTO 2001. Springer; 2001. p. 41–62.
16. Boldyreva A, Goyal V, Kumar V. Identity-based encryption with efficient revocation. In: Proceedings of the 15th ACM conference on Computer and communications security. ACM; 2008. p. 417–426.

17. Boldyreva A, Goyal V, Kumar V. Adaptive-ID Secure Revocable Identity-Based Encryption. In: Proceedings of the 15th ACM conference on Computer and communications security. ACM; 2008. p. 417–426.
18. Boldyreva A, Goyal V, Kumar V. Constructions o f CCA-Secure Revo cable Identity-Based Encryption. In: Proceedings of the 15th ACM conference on Computer and communications security. ACM; 2008. p. 417–426.
19. Boldyreva A, Goyal V, Kumar V. An Efficient and Provable Secure Revocable Identity-Based Encryption Scheme. In: Proceedings of the 15th ACM conference on Computer and communications security. ACM; 2008. p. 417–426.
20. Lee K, Lee DH, Park JH. Efficient revocable identity-based encryption via subset difference methods. Designs, Codes and Cryptography. 2017; 85(1):39–76. https://doi.org/10.1007/s10623-016-0287-3
21. Watanabe Y, Emura K, Seo JH. New revocable IBE in prime-order groups: Adaptively secure, decryption key exposure resistant, and with short public parameters. In: Cryptographers Track at the RSA Conference. Springer; 2017. p. 432–449.
22. Lee K, Choi SG, Lee DH, Park JH, Yung M. Self-updatable encryption: Time constrained access control with hidden attributes and better efficiency. In: International Conference on the Theory and Application of Cryptology and Information Security. Springer; 2013. p. 235–254.