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

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 satisfying 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.
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 identity 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 ∈ 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 ∈ 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 ∈ 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 trivial 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 threefold:

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 challenge 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 satisfying $ID_{j} \notin \text{Prefix}(ID_{h})$, 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)  | ×                    | ×                   | ×           | DBDH          |
| Seo(2015)  | ×                    | ×                   | √           | q-Type        |
| Seo(2015)  | ×                    | ×                   | √           | static        |
| Ryu(2015)  | √                    | ×                   | √           | q-RW2         |
| Lee(2016)  | ×                    | ×                   | √           | q-Type        |
| Lee(2016)  | ×                    | √                   | √           | static        |
| Xing(2016) | √                    | ×                   | √           | 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 \( \log N \) 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 \( (\log N)^2 \) subkeys since for every subkey of his parent he needs to generate \( \log N \) subkeys respectively, which results to \( (\log N)^l \) 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 \( (l\log N) \)-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 \( (l \cdot \log N) \)-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 \( (\log N) \) and the size of the update key is \( (l + r\log(N/r)) \). 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)  | \((h + 2)r\) | \(O(l)\) | \((l + 4)G\) | \(O(l^2 \log N)\) | \(O(\log(N/r))\) |
| Seo(2015)  | \(3r\)    | \(O(1)\) | \((l + 6)G\) | \(O(\log N)\) | \(O(\log(N/r))\) |
| Seo(2015)  | \(3r\)    | \(O(1)\) | \((l + 6)G + G_t\) | \(O(\log N)\) | \(O(\log(N/r))\) |
| Ryu(2015)  | \((2h + 2)r\) | \(O(h)\) | \(7G + G_t\) | \(O(\log N)\) | \(O(h\log(N/r))\) |
| Lee(2016)  | \((2h + 3)r\) | \(O(l)\) | \(6G + G_t\) | \(O(\log N)\) | \(O(l + r\log(N/r))\) |
| Lee(2016)  | \(4r\)    | \(O(l)\) | \((l + 5)G + G_t\) | \(O(\log N)\) | \(O(l + r\log(N/r))\) |
| Xing(2016) | \((3h + 2)r\) | \(O(h)\) | \(7G + G_t\) | \(O(\log N)\) | \(O(h\log(N/r))\) |
| Our RHIBE  | \((4h + 4)r\) | \(O(h)\) | \(9G + G_t\) | \(O(\log N)\) | \(O(h + \log(N/r))\) |

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 been considered by Seo [14], which discusses about the case where several decryption keys \( dk, T \) for the target identity \( I' \) are leaked to an adversary but the target decryption key \( dk, T \) 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 \( \pi = (\text{Setup, GenKey, DeriveKey, UpdateKey, Encrypt, Decrypt, Revoke}) \) as following:

1. **Setup**(1\( ^{\lambda} \)): It takes a security parameter \( \lambda \), 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 definition by all the bounded RHIBEs.

2. **GenKey**(ID\(_{k} \), ST\(_{ID_{k}}, T \), PP): This algorithm takes as input ST\(_{ID_{k}}, T \) and an identity ID\(_{k} \) outputs the secret key SK\(_{ID_{k}} \) and updates ST\(_{ID_{k}, T} \).

3. **UpdateKey**(T, RL\(_{ID_{k}}, T \), DK\(_{ID_{k}, T} \), ST\(_{ID_{k}, T} \), PP): This algorithm takes as input the revocation list RL\(_{ID_{k}, T} \), state information ST\(_{ID_{k}, T} \), the decryption key DK\(_{ID_{k}, T} \), and a time period T. Then, it outputs the update key UK\(_{ID_{k}, T} \).

4. **DeriveKey**(SK\(_{ID_{k}}, UK_{ID_{k}, T} \), PP): This algorithm takes as input SK\(_{ID_{k}} \), of ID\(_{k} \) and UK\(_{ID_{k}, T} \), 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 \( \bot \).

5. **Encrypt**(ID\(_{k} \), T, M, PP): This algorithm takes as input a message M, ID\(_{k} \), and the current time T and outputs the ciphertext CT.

6. **Decrypt**(CT\(_{ID_{k}, T} \), DK\(_{ID_{k}, T} \), PP): This algorithm takes as input CT\(_{ID_{k}, T} \) and DK\(_{ID_{k}, T} \), and outputs the message if ID\(_{k} \) is a prefix of ID\(_{l} \) and \( TT = T' \), else outputs \( \bot \).
7. **Revoke**\(_{(RL_{ID|_k}, ST_{ID|_k}, ID|_k}, T)\): This algorithm takes as input ID|\(_k\) and T, updates RL\(_{ID|_{k-1}}\) managed by ID|\(_k\), 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.

\[
\operatorname{Exp}_{\mathcal{A}}^{\text{IND-RID-CPA}}(\lambda) :
\]

\[
(MK, PP, RL_e, ST_e) \leftarrow \text{Setup}(\lambda);
(M'_0, M'_1, ID|_k^0, T^0, ST) \leftarrow \mathcal{A}^O(\text{Find}, PP);
\]

\[
b \leftarrow \{0, 1\}; CT' \leftarrow \text{Encrypt}(PP, ID|_k^b, T', M'_b);
\]

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

In the above experiment, \(O\) is a set of oracles \(\{\text{SKGen}_Q(\cdot), \text{KeyUp}_Q(\cdot, \cdot), \text{Revoke}_Q(\cdot, \cdot), \text{DKGen}_Q(\cdot, \cdot)\}\) defined as follows:

- **SKGen\(_Q(\cdot)\):** For \(ID|_k \in \mathcal{T}^k\), it returns \(SK_{ID|_k}\) (by running GenKey(ID|\(_k\), ST\(_{ID|_{k-1}}\), PP) → 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) → KU\(_{ID|_{k-1}}\)).

- **Revoke\(_Q(\cdot, \cdot)\):** For \(ID|_k \in \mathcal{T}^k\) 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 \(ID|_k \in \mathcal{T}^k\) and \(T \in \mathcal{T}\), it returns DK\(_{ID|_k}\) (by running DeriveKey(SK\(_{ID|_k}\), UK\(_{ID|_k}\), T, PP) → DK\(_{ID|_k}\)).

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

1. **Revoke\(_Q(\cdot, \cdot)\)** can be queried on time \(T\) if **KeyUp\(_Q(\cdot, \cdot)\)** was queried on \(T\).
2. **DKGen\(_Q(\cdot, \cdot)\)** cannot be queried on time \(T\) before **KeyUp\(_Q(\cdot, \cdot)\)** was queried on \(T\).
3. If \(\mathcal{A}\) requested a private key query for \(ID|_k^i\) that is a prefix of \(ID|_k\) where \(k \leq l\), then the identity \(ID|_k^i\) or one of its ancestors should be revoked at some time \(T\) where \(T \leq T'\).
4. \(\mathcal{A}\) cannot request a decryption key query for the challenge identity \(ID|_k^*|\) 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\(_Q(\cdot, \cdot)\)** and **Revoke\(_Q(\cdot, \cdot)\)** for same identity in increasing order of time.

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

### 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 \(p = p_1 \times p_2 \times p_3, p_1, p_2, p_3\) 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 \(Adv_{\mathcal{A}}^{\text{RHIBE}}(\lambda) = |Pr[A(D, T_1)] - Pr[A(D, T_2)]|\) for any PPT algorithm \(\mathcal{A}\) and parameters \(D, T_1, T_2\).
Assumption 1. Let $g^e \in G_p$, $D = (G, g)$, $T_1, e \in G_p$, $T_2, e \in G_p$, we say that $G$ satisfies Assumption 1 if $Adv_{G, \lambda}(A)$ is a negligible function of $\lambda$ for any PPT algorithm is $A$.

Assumption 2. Let $g^e \in G_p$, $g_1, X_1, Y_1, Y_2, z^e \in G_p$, $g_1, X_1, Y_1, Y_2, z^e \in G_p$, $x, s \in \mathbb{Z}_n$, $T_1$ be $e(g, g_1^x)$, $T_2, e \in G_p$, $D = (G, g, g_1, \gamma = x, s \in \mathbb{Z}_n, T_1$ be $e(g, g_1^x)$, $T_2, e \in G_p$, we say that $G$ satisfies Assumption 2 if $Adv_{G, \lambda}(A)$ is a negligible function of $\lambda$ for any PPT algorithm is $A$.

Assumption 3. Let $g^e \in G_p$, $g_1, X_1, Y_1, Y_2, z^e \in G_p$, $g_1, X_1, Y_1, Y_2, z^e \in G_p$, $x, s \in \mathbb{Z}_n$, $T_1$ be $e(g, g_1^x)$, $T_2, e \in G_p$, $D = (G, g, g_1, \gamma = x, s \in \mathbb{Z}_n, T_1$ be $e(g, g_1^x)$, $T_2, e \in G_p$, we say that $G$ satisfies Assumption 3 if $Adv_{G, \lambda}(A)$ is a negligible function of $\lambda$ for any PPT algorithm is $A$.

Assumption 4. Let $g^e \in G_p$, $g_1, X_1, Y_1, Y_2, z^e \in G_p$, $g_1, X_1, Y_1, Y_2, z^e \in G_p$, $x, s \in \mathbb{Z}_n$, $T_1$ be $e(g, g_1^x)$, $T_2, e \in G_p$, $D = (G, g, g_1, \gamma = x, s \in \mathbb{Z}_n, T_1$ be $e(g, g_1^x)$, $T_2, e \in G_p$, we say that $G$ satisfies Assumption 4 if $Adv_{G, \lambda}(A)$ is a negligible function of $\lambda$ for any PPT algorithm is $A$.

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 $GS = ((N = p_1, p_2, p_3, G, G_T, e), g, g_1, g_3) \leftarrow \mathcal{G}(\lambda)$ be the bilinear group, where $\lambda$ is a security parameter and $g_3$ denotes a generator of $G_p$, $g_3$ denotes a generator of $G_p$, and $g$ be a generator of $G_p$.

3.1 HIBE scheme

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

$$\kappa(I, y, r) = \langle w^v, g^v, g^y, (u'h)^v \rangle$$

and an expression $g^x \kappa(I, y, r)$ as

$$g^x \kappa(I, y, r) = \langle g^x w^v, g^v, g^y, (u'h)^v \rangle$$

HIBE.Setup($GS$): It selects $u, h, w, v \in \mathbb{Z}_p$, and $x \in \mathbb{Z}_p$. It outputs a master key $MK = \alpha$ and public parameters $PP = ((u, h, w, v), e)$. It outputs a master key $MK = \alpha$ and public parameters $PP = ((u, h, w, v), e)$.

HIBE.GenKey($ID_i, MK, PP$): Let the identity $ID_i = \langle I_1, \ldots, I_j \rangle \in \mathbb{Z}_p$, and $I = \{0, 1\}^i$ be the identity space. It chooses $\lambda_1, \ldots, \lambda_k, r_1, \ldots, r_k, \gamma_1, \ldots, \gamma_k \in \mathbb{Z}_p$ where $\lambda_1 + \cdots + \lambda_k = \alpha$ and outputs a private key $SK_{ID_i} = \{G_0^{\gamma_1}, G_1^{\gamma_1}, \ldots, G_{k-1}^{\gamma_1}\}$.

HIBE.RandKey($ID_i, SK_{ID_i}, PP$): Let $SK_{ID_i} = \{G_0^{\gamma_1}, G_1^{\gamma_1}, \ldots, G_{k-1}^{\gamma_1}\}$. It outputs a re-randomized private key $SK_{ID_i} = \{G_0^{\gamma_1}, G_1^{\gamma_1}, \ldots, G_{k-1}^{\gamma_1}\}$.

HIBE.Delegate($ID_i, SK_{ID_i}, PP$): Let $SK_{ID_i} = \{G_0^{\gamma_1}, G_1^{\gamma_1}, \ldots, G_{k-1}^{\gamma_1}\}$. It outputs a temporal delegate private key $SK_{ID_i} = \{G_0^{\gamma_1}, \gamma_1, \gamma_2, \gamma_{k-1} \}$. Next, it outputs a delegated private key $SK_{ID_i} = \{G_0^{\gamma_1}, \gamma_1, \gamma_2, \gamma_{k-1} \}$.

HIBE.Encaps($ID_i, s, PP$): Let $ID_i = \langle I_1, \ldots, I_j \rangle \in \mathbb{Z}_p$. It chooses $t_1, \ldots, t_j \in \mathbb{Z}_p$ and outputs a ciphertext $CT_{ID_i} = \langle g^x, (w^v)^{t_1}, (u'h)^{t_1}, g^y \rangle$ and a session key $EK = \Omega'$. Otherwise, it outputs $\bot$.
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'_0, K'_1, K'_2, K'_3\}_{j=1}^k). It chooses \(\lambda_1, \ldots, \lambda_k \overset{\$}{\leftarrow} Z_p\) where \(\lambda_1 + \cdots + \lambda_k = \delta\) and sets TSK = (\{K_0^{T_0}, K_{1,T}, K_{2,T}, K_{3,T}\}_{j=1}^k). It outputs a new private key SK_{ID_k} ← HIBE.RandKey(ID_k, TSK_{ID_k}, PP).

**HIBE.MergeKey(SK^{(1)}_{ID_k}, SK^{(2)}_{ID_k}, \eta, PP):** Let SK^{(1)}_{ID_k} = (\{K'_0, K'_1, K'_2, K'_3\}_{j=1}^k) and SK^{(2)}_{ID_k} = (\{K''_0, K''_1, K''_2, K''_3\}_{j=1}^k) be two private keys for the same identity ID_k. It computes a temporal private key TSK = (\{K_{0,T}, K_{1,T}, K_{2,T}, K_{3,T}\}_{j=1}^k). Next, it outputs a merged private key SK_{ID_k} ← HIBE.ChangeKey(TSK, \eta, PP). Note that the master key parts are \(\alpha_1 + \alpha_2 + \eta\) if the master key parts of SK^{(1)}_{ID_k} and SK^{(2)}_{ID_k} are \(\alpha_1\) and \(\alpha_2\) respectively.

### 3.2 IBE scheme

A trivial extension to RHIBE from the HIBE in [12] constructs the decryption key of \((T, ID_k)\) as \(D_0 = g^{\kappa^0\kappa(T, y, r)}\) and \(D_i = g^{\kappa^i\kappa(I, y, r)}\), \(i = 1 \cdots k\). It outputs a master key MK = \(\beta\) and public parameters PP = (\((p, G, G_T, \epsilon), g, u_0, h_0, w_0, \omega, \Omega = e(g, g)^\alpha\)). It chooses a time \(T\) and the master key MK, and the public parameters PP. It outputs a ciphertext CT_T = \((C_0 = g^r, C_1 = w_0^g v_0, C_2 = u_0^g h_0, C_3 = g^r)\) and the session key EK = \(\Omega\).

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

**HIBE.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 = \(e(C_0, K_0)e(C_1, K_1)(e(C_1, K_1) e(C_2, K_2))\), otherwise, it outputs ⊥.

The construction 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 \(BT\) of depth \(n\). Each user is assigned to a different leaf node in \(BT\). The collection \(S\) is defined as \(\{S_i\}\) where \(S_i\) is the set of all leaves in a subtree \(T_i\) with a subroot \(v_i \in BT\). It outputs the full binary tree \(BT\).

**CS.Assign(BT, ID):** Let \(v_0\) be a leaf node of \(BT\) that is assigned to the user \(ID\). Let \((v_{k_0}, v_{k_1}, \ldots, v_{k_l})\) be the path from the root node \(v_0 = v_0\) to the leaf node \(v_{k_0} = v_{ID}\). For all \(j \in [k_0, \ldots, k_l]\), it adds \(S_i\) into \(PV_{ID}\). It outputs the private set \(PV_{ID} = |S_i|\).
CS.Cover\((BT, R)\): It first computes the Steiner tree \(ST(R)\). Let \(T_{k_1}, \ldots , T_{k_n} \) be all the sub-

trees of \(BT\) that hang off \(ST(R)\), that is all subtrees whose roots \(v_{k_i}, \ldots , v_{k_n}\) are not in \(ST(R)\) but

adjacent to nodes of outdegree 1 in \(ST(R)\). For all \(i \in \{k_1, \ldots , k_n\}\), 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 ⊥.

3.4 Construction

RHIBE.Setup\((\lambda, N_{max})\): The Setup algorithm takes a security parameter \(\lambda\) and a maximum

number of users for each level \(N_{max}\) as input. It firstly runs \(G\) to obtains two groups \(G, G_T\) of

order \(p = p_1 \cdot p_2 \cdot p_3\), 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 \(g, g_2, g_3\) denote the generators of \(G_{p_1}, G_{p_2}\), and \(G_{p_3}\),

in order. It selects a random exponent \(\alpha \in Z_p^\ast\) set \(\Omega = e(g, g)^\alpha\). 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\((ID|k, ST_{ID|k}, PP)\): This algorithm takes as input an identity

\(ID|k = (I_1, \ldots , I_k) \in T^k\), the state \(ST_{ID|k}\), which contains \(BT_{ID|k}\),

1. If \(ST_{ID|k}\), is empty, it obtains \(BT_{ID|k} \leftarrow CS.Setup(N_{max})\) and then it sets \(ST_{ID|k} = (BT_{ID|k}, \beta_{ID|k}, z_{ID|k})\),

where \(\beta_{ID|k}\) is a false master key and \(z_{ID|k}\) is a PRF key.

2. It first assigns \(ID|k\) to a random leaf node \(v \in BT_{ID|k}\), and obtains a node set \(Path(ID|k) \leftarrow \)

\(CS.Assign(BT_{ID|k}, ID|k)\) for \(ID|k\). For each \(S_0 \in Path\), it computes \(\gamma_0 = PRF(z_{ID|k}, L_0)\)

where \(L_0 = Label(S_0)\) and obtains an HIBE private key \(SK_{HIBE,S_0} \leftarrow HIBE.GenKey(ID|k, \gamma_0, PP)\).

Finally, it outputs a private key \(SK_{ID|k} = (Path, \{SK_{HIBE,S_i} | S_i \in Path\})\). Note that the master key part

of \(SK_{HIBE,S_i}\) is \(\gamma_0\).

RHIBE.UpdateKey\((T, RL_{ID|k}, DK_{ID|k}, ST_{ID|k}, PP)\): let \(DK_{ID|k,T} = (RSK_{HIBE,ID|k}, RSK_{IBE,T})\),

the state \(ST_{ID|k} = (BT_{ID|k}, \beta_{ID|k}, z_{ID|k})\) with \(k \geq 1\)

1. It first obtains a randomized decryption key \(RDK_{ID|k,T}\) as \((RSK_{HIBE, RSK_{HIBE}}) \leftarrow RHIBE.

RandDK(DK_{ID|k,T}, -\beta_{ID|k}, PP)\).

2. It derives the set of revoked identities \(R\) at time \(T\) from \(RL_{ID|k}\). Next, it obtains a covering set \(CV_R = \{S_i\}\) by running \(CS.Cover(BT_{ID|k}, R)\).

3. For each \(S_i \in CV_R\), it computes \(\gamma_i = PRF(z_{ID|k}, L_i)\) where \(L_i = Label(S_i)\) and obtains an IBE private key \(SK_{IBE,S_i} \leftarrow IBE.GenKey(T, -\gamma_i, PP)\). Then it computes

\(SK_{IBE,S_i} \leftarrow IBE.MergeKey(SK_{IBE,S_i}, RSK_{IBE, \beta_{ID|k}, PP})\)

4. It finally outputs an update key \(UK_{ID|k,T} = (CV_R, \{SK_{IBE,S_i}, RSK_{HIBE} | S_i \in CV_R\})\). Note that the master key parts of \(RSK_{HIBE}\) and \(SK_{IBE,S_i}\) are \(\eta\) and \(\alpha - \eta\) for some random \(\eta\)

respectively.

RHIBE.DeriveKey\((SK_{ID|k}, UK_{ID|k,T}, PP)\): This algorithm takes as input a private key

\(SK_{ID|k} = (Path, \{SK_{HIBE,S_i}, RSK_{HIBE} | S_i \in Path\})\) for an identity \(ID|k\), an update key

\(UK_{ID|k,T} = (CV_R, \{SK_{IBE,S_i}, RSK_{HIBE,ID|k}, S_i \in CV_R\})\) for time \(T\).

1. If \(K = 0\), then \(SK_{ID|k} = MK = \alpha\) and \(UK_{ID|k,T}\) is empty. It selects a random exponent \(\eta \in Z_p\). It

then obtains \(RSK_{HIBE, ID|k} \leftarrow HIBE.GenKey(ID|k, \eta, PP)\) and \(RSK_{IBE,T} \leftarrow IBE.GenKey(T, \alpha - \eta, PP)\). It outputs a decryption key \(DK_{ID|k,T} = (RSK_{IBE, T}, RSK_{HIBE, ID|k})\).
2. If \( k \geq 1 \), then if \( ID[j] \notin RL_{ID_{k-1}} \), then it obtains \((S_a, S_i)\) by running \texttt{CS.Match}(\texttt{CV}_k, \texttt{Path}). Otherwise, it outputs \( \perp \). It derives \( SK_{HIBE_S} \) from \( SK_{ID_k} \) and \( SK_{HIBE_E} \) from \( UK_{ID_{k-1}} \).

3. It obtains \( RSK^e_{HIBE.ID_k} \leftarrow HIBE.Delegate(ID[j], RSK^e_{HIBE.ID_{k-1}}, PP) \) since \( ID[j] \in \text{Prefix}(ID) \).

   Next, it selects a random exponent \( \eta \in Z_p \) obtains \( RSK_{HIBE.ID_k} \leftarrow HIBE.MergeKey(RSK^e_{HIBE.ID_k}, SK_{HIBE_S}, \eta, PP) \) and obtains \( RSK_{IBE.T} \leftarrow IBE.ChangeKey(SK_{HIBE_S}, -\eta, PP) \) 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 \( \eta' \) and \( \alpha - \eta' \) for some random \( \eta' \) respectively.

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

   \textbf{RHIBE.CT}(\( ID[j], T, M, PP \)): This algorithm takes as input an identity \( ID[j] = (I_1, \ldots, I_l) \in T \), time \( T \), a message \( M \in M \). It chooses a random exponent \( t \in Z_p \). Next it obtains \((CH_{HIBE.ID_k}, EK_{HIBE}) \leftarrow HIBE.Encaps(\( ID[j], t, PP \)). It also obtains \((CH_{IBE.T}, EK_{IBE}) \leftarrow IBE.Encaps(T, t, PP) \). It outputs a ciphertext \( CT_{ID_k,T} = (CH_{IBE.T}, CH_{HIBE.ID_k}, C = \Omega^t \cdot M) \).

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

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

### 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

\[
(g^{t+\Pi_i h_i} w_i^h, g^t, (u_i^h h_i)^{\alpha_i} v_i^h, \{g^{h_i} w_i^h, g^{u_i} (u_i^h)^{\gamma_i} v_i^h\}_i)
\]

The RHIBE.Decrypt algorithm takes \( CT_{ID_k,T} \) as input, where

\[
CT_{ID_k,T} = (g^t, w_i^h, (u_i^h h_i)^{\gamma_i}, \{w^i v^i, (u_i^h)^{\gamma_i} v^i\}_i),
\]

and computes \( B = C/M \) as

\[
B = \frac{e(g^t, g^{w_i^h} \Pi_i h_i w_i^h) e(g^{h_i} w_i^h, (u_i^h h_i)^{\gamma_i} v_i^h)}{e(w_i^h v_i^h, g^t) e((u_i^h)^{\gamma_i} v_i^h, g^t)} \prod_{i=1}^{k} \frac{e(g^{v_i}, (u_i^h)^{\gamma_i} v_i^h)}{e(w^i v^i, g^t) e((u_i^h)^{\gamma_i} v_i^h, g^t)} = \Omega^t
\]

### 4 Security analysis

We use the dual system encryption proof technique to prove the adaptive security of our U-RHIBE. We adopt the concept of \textit{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.
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 B is initially given random elements \( g, u, v, w, u_0, v_0, w_0 \in G_p, g_2 \in G_p^2 \), \( g_3 \in G_p \), as well as random exponents \( \psi_1, \psi_2, a_1, a_2, a', b', s, \delta_1, \delta_2, \gamma \).

We define the ephemeral semi-functional ciphertext and five types of ephemeral semi-functional ciphertexts of a normal ciphertext \( CT_{ID, T} \) by changing the \( C_0 \) element into \( G_{p_2} \) and the \( l+1 \) numbers of the ciphertext-element-groups \( \langle C_{i_1}, C_{i_2}, C_{i_3} \rangle \) into different types. The definitions of ephemeral semi-functional ciphertexts called ESF-1-CT, ESF-2\(^k\)-CT, ESF-3\(^k\)-CT, ESF-4\(^k\)-CT, CT and ESF-5-CT where \( 0 \leq k \leq l \) are in Appendix.A. In the definitions of the semi-functional ciphertext, we add \( G_p \) term on the first element of all ciphertext-element-groups.

RHIBE.EncryptSF: It firstly obtains the normal ciphertext \( CT_{ID, T} = \langle C, C_0, \{C_{i_1}, C_{i_2}, C_{i_3} \}_{i=0}^l \rangle \) for an identity \( ID = \langle I_1, \ldots, I_k \rangle \in T^t \), a time \( T \in T \) and a message \( M \in M \). It chooses exponents \( \gamma, \delta_1, \delta_2 \in Z_p \) and outputs the SF-CT \( \overline{CT_{ID, T}} \) as

\[
\langle C, C_0 \cdot g_2^\gamma, C_{i_1} \cdot g_2^{\delta_1}, C_{i_2} \cdot g_2^{\delta_2}, C_{i_3} \cdot g_2, \{C_{i_4}, C_{i_5}, C_{i_6} \}_{i=7}^l \rangle
\]

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 small 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 definition 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 definition of the semi-functional secret key, we add \( G_{p_3} \) term on the first 2 elements and the last element of the last element-group.

RHIBE.KeySF \( (ID, ST_{ID, T}, PP, \theta) \rightarrow \overline{SK_{HIBE, S}} \): It constructs the correlative sub-key \( SK_{HIBE, S} = \langle \{K_{i_1}, K_{i_2}, K_{i_3} \}_{i=1}^l \rangle \) to the node \( \theta \in \text{Path}(ID) \) in the BT \( \text{ID,} \text{T}_{i=1}^l \) as follows: It chooses random exponents \( \psi', r \in Z_p \) and choose \( a_1, \psi_1 \in Z_p \), then it constructs \( \chi^\psi(I_p, \psi', r) \) for the last element-group as

\[
(\omega' (g, g_2) y^{\psi_1}, g' (g, g_2) y^{\psi}, g' (g, g_2) y^{\psi_1} (u^h y))
\]

And the construction of the other element-groups follows the construction of \( SK_{HIBE, S} \) in RHIBE.GenKey.

We define one type of semi-functional update key and five types of ephemeral semi-functional update key. The definition 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.
We define a sequence of games to verify the advantage in distinguishing the security against Type-2 adversary.

### 4.2 Sequence of games

- **RHIBE.UpdateKeySF** ($T, ST_{ID_{Rk-1}}, RL_{ID_{Rk-1}}, r, PP, \theta$) $\rightarrow \overline{TUK}$: It constructs the correlative component key $TU_{K_{ID_{Rk-1}, r, \theta}} = \{(U_{0,0}, U_{1,1}, U_{1,2}, U_{1,3})_{i=0}^{k-1}\}$ to the node $\theta \in KUNode$ as follows: It chooses random exponents $y', r \in Z_p$ and chooses $\sigma_2, \psi_2 \in Z_p$, then it constructs $K_y(T, y', r)$ of the first element-group $\{(U_{0,0}, U_{1,1}, U_{1,2}, U_{1,3})_{i=0}^{k-1}\}$ as

$$\left( w_0'(gg_0)^{y'+\sigma_2}, g'g_0^{y'}, \psi_0'g_0^{r+\psi_2}U_0, (u_0^y h_0) \right)$$

And the contraction of the other element-groups follows the construction of $RSK_{HIBE}$ and $SK_{HIBE, S_{Rk}}$ in RHIBE.UpdateKey.

- **RHIBE.DeriveKeySF**: Let $SK_{ID_{Rk}}$ be a semi-functional secret key generated by the RHIBE. GenKeySF algorithm and $\overline{UK}_{ID_{Rk+1}, r}$ be a semi-functional update key for time $T$ generated by the RHIBE.UpdateKeySF algorithm. If $ID[k] \notin RL_{ID_{Rk-1}}$, then it finds a unique node $\theta$ by running $CSMatch(CV_{Rk}(BT_{ID_{Rk-1}}, RL_{ID_{Rk-1}}, T), Path(ID[k]))$. Otherwise, it outputs $\perp$. It derives $PSK_{\theta'} = \{(K_0, K_{1,1}, K_{1,2}, K_{1,3})_{i=1}^{k-1}\}$ from $SK_{ID_{Rk}}$ and $TU_{K_{\theta'}} = \{(U_{0,0}, U_{1,1}, U_{1,2}, U_{1,3})_{i=0}^{k-1}\}$ from $\overline{UK}_{ID_{Rk+1}, T}$ for the node $\theta'$. Then the semi-functional decryption key $\overline{DK}_{ID_{Rk+1}}$ is

$$\overline{DK}_{ID_{Rk+1}} = \{(D_{0,0}, D_{1,1}, D_{1,2}, D_{1,3})_{i=0}^{k-1}\}$$

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

### 4.2.1 Game definitions

We define a sequence 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 ciphertext 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 secret keys, update keys, decryption keys and challenge ciphertext 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 $G_{C'}$, 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.

- **$G_{ES}$**: 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.

- **$G_{ES'}$**: All secret keys, update keys, and challenge ciphertext are changed to semi-functional. The decryption keys are still normal.
The decryption keys are changed to ESF-2. The other keys and the challenge ciphertext are still semi-functional.

**GESF**: 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.

**SF0**: 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 **SF**, except for a added restriction about the challenge key identity vector. We explain the restriction in Sec. 4.6.

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

Let $Adv^{{\mathcal{R}}_{\text{HIBE}}}({\mathcal{A}})$ be the advantage of $\mathcal{A}$ in the real game. From the all the lemmas in this section, we obtain the following equation

$$Adv^{{\mathcal{R}}_{\text{HIBE}}}({\mathcal{A}}) \leq |Adv^{{\mathcal{G}_{\text{Final}}}}({\mathcal{A}}) - Adv^{{\mathcal{G}_{\text{C}}}}({\mathcal{A}})| + |Adv^{{\mathcal{G}_{\text{C}}}}({\mathcal{A}}) - Adv^{{\mathcal{G}_{\text{ESF}}}}({\mathcal{A}})|$$

$$+ |Adv^{{\mathcal{G}_{\text{ESF}}}}({\mathcal{A}}) - Adv^{{\mathcal{G}_{\text{SF}}}}({\mathcal{A}})| + |Adv^{{\mathcal{G}_{\text{SF}}}}({\mathcal{A}}) - Adv^{{\mathcal{G}_{\text{Final}}}}({\mathcal{A}})|$$

$$\leq Adv^{A_1}_B(\lambda) + Adv^{A_2}_B(\lambda)$$

$$+ (O(q_n \log N_{\text{max}}) + O(q_n r_{\text{max}} \log N_{\text{max}}) + O(I))(Adv^{A_3}_B + Adv^{A_4}_B)$$

### 4.3 Definition of oracles

We introduce seven oracles which answer queries from the challenger $B$ by sampling various distributions of group elements from a composite order bilinear group. The outputs of Oracle...
O₁ 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 \( p = p₁p₂p₃ \) and initially choose random elements \( g, u, v, w, \omega₀, \omega₁, \omega₂ \in G_{p₁}, g₂ \in G_{p₂}, g₃ \in G_{p₃} \) as well as random exponents \( ψ₁, ψ₂, α₁, α₂, α', β', s, δ₁, δ₂, γ \in \mathbb{Z}_n \). They provide the attacker with a description of the group \( G \), as well as the group elements

\[
g, u, v, w, g'g₂, \omega₀g'g₂, \omega₁g'g₂, \omega₂g'g₂, g'v'g₂, v'g'g₂, \omega₀g'g₂, v₀g'g₂, g'g₂, v₀g'g₂, g', v'
\]

(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 \( O₀ \) to \( O₅ \) in which the simulators will be allowed to produce a normal challenge decryption key. The outputs of Oracle \( O₀ \) will allow a simulator to produce a semi-functional challenge ciphertext, a normal challenge (H)IBE private key. The outputs of Oracle \( O₁ \) 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 \( O₂ \) 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 \( O₃ \) 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 \( O₄ \) will allow a simulator to produce a semi-functional challenge ciphertext, and a semi-functional challenge decryption key.

We define the oracles from \( O₅ \) to \( O₇ \) in which the simulators will be allowed to produce a semi-functional challenge (H)IBE key. The outputs of Oracle \( O₅ \) will allow a simulator to produce a semi-functional ciphertext, and an ephemeral semi-functional challenge decryption key. The outputs of Oracle \( O₆ \) will allow a simulator to produce an ephemeral semi-functional (ESF-1) ciphertext, and a type-2 ephemeral semi-functional (ESF-2) challenge decryption key. Finally, the outputs of Oracle \( O₇ \) will allow a simulator to produce a semi-functional ciphertext, and a semi-functional challenge decryption key.

**Oracle** \( O₀ \). The first oracle, which we will denote by \( O₀ \), responds to queries as follows. Upon receiving a challenge HIBE-key-type query for \( I \in \mathbb{Z}_n \), it chooses \( r, y' \in \mathbb{Z}_n \) randomly and returns the group elements

\[
(w', g', v'(u'h)', g')
\]

(3)
to the attacker. Upon receiving a challenge IBE-key-type query for \( T \in \mathbb{Z}_m \), it chooses \( r', y' \in \mathbb{Z}_m \) randomly and returns the group elements

\[
(w'_0, g'_0, v'_0(u'_0h_0)', g')
\]

(4)
to the attacker. Upon receiving a challenge decryption-key-type query for \( I \in \mathbb{Z}_n \) and \( T \in \mathbb{Z}_m \), it chooses \( r, y', r', y' \in \mathbb{Z}_n \) randomly and returns the group elements

\[
(w', g', v'(u'h)', g', w'_0, g'_0, v'_0(u'_0h_0)', g')
\]

(5)
to the attacker. Upon receiving a ciphertext-type query for \( I \in \mathbb{Z}_m \), it chooses \( t \in \mathbb{Z}_n \) randomly and returns the group elements

\[
(w'g'_0, v', g', (u't'h)')
\]

(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

$$ (w_0^{e^{t_0}}, v^{e^{t_0}}, (u_0^{e^{t_0}}h_0)^{b_0} ) $$

(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 $X_2$, $Y_2 \in G_{p_1}$, $X_3$, $Y_3 \in G_{p_1}$, randomly. It returns the group elements

$$ (w^{r''}, g^{r''}, v^{r''} (u'h)^{r''} X_2 X_3, g^{r''} Y_2 Y_3 ) $$

(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 $O_2$** The next oracle, which we will denote by $O_2$, responds to queries as follows.

Upon receiving a challenge HIBE-key-type query for $T \in Z_n$, it chooses $r''$, $y'' \in Z_n$ randomly, and also chooses $X_2$, $Y_2 \in G_{p_1}$, $X_3$, $Y_3 \in G_{p_1}$, randomly. It returns the group elements

$$ (w_0^{r''}, g^{r''}, v^{r''} (u_0^{e^{t_0}}h_0)^{r''} X_2 X_3, g^{r''} Y_2 Y_3 ) $$

(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 $O_3$** 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 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

$$ (w_2^{e^{t_0}}v_2^{e^{t_0}}g_2^{e^{t_0}}, g_2^{e^{t_0}}, (u_0^{e^{t_0}}h_0)^{e_0 (d'T''+b_1)} ) $$

(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

$$ (w_2^{e^{t_0}}v_2^{e^{t_0}}g_2^{e^{t_0}}, g_2^{e^{t_0}}, (u_0^{e^{t_0}}h_0)^{e_0 (d'T''+b_1)} ) $$

(11)

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 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

$$ (w_2^{e^{t_0}}v_2^{e^{t_0}}g_2^{e^{t_0}}, g_2^{e^{t_0}}, (u_0^{e^{t_0}}h_0)^{e_0 (d'T''+b_1)} ) $$

(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

$$ (w_2^{e^{t_0}}v_2^{e^{t_0}}g_2^{e^{t_0}}, g_2^{e^{t_0}}, (u_0^{e^{t_0}}h_0)^{e_0 (d'T''+b_1)} ) $$

(13)

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 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 $X_2$, $Y_2 \in G_{p_2}$, $X_3$, $Y_3 \in G_{p_2}$, randomly. It returns the group
randomly and returns the group elements
elements
\[(w'_0, g'_{\mu}, v_0' (u'_{\mu} h_0)' \cdot X_2 X_3, g' \cdot Y_2 Y_3)\] (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 \(I \in \mathbb{Z}_m\) by choosing \(r, y' \in \mathbb{Z}_n\) randomly and returns the group elements
\[(w' \cdot (g, g_3)^{y'^*}, g' \cdot (g, g_3)^{v'^*} (u'h)' \cdot g')\] (15)
to the attacker. Upon receiving a challenge IBE-key-type query for \(T \in \mathbb{Z}_n\), it chooses \(r', y'' \in \mathbb{Z}_n\) randomly and returns the group elements
\[(w''_0, g'' (g, g_3)^{y''*}, g'' (g, g_3)^{v''*} (u''_{\mu} h_0)' \cdot g')\] (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 \(I, T \in \mathbb{Z}_m\), it chooses \(r, y', r', y'' \in \mathbb{Z}_n\) randomly, and also chooses \(X_2, Y_2, X_3, Y_3, Y'_3 \in G_{p_2}, X_3, Y_3, Y'_3 \in G_{p_3}\) randomly. It returns the group elements
\[(w', g', v' (u'h)' \cdot X_2 X_3, g' \cdot Y_2 Y_3, w'_0, g'' (g, g_3)^{y''*}, v'_0 (u''_{\mu} h_0)' \cdot X_2 X_3, g'' \cdot Y_2 Y'_3)\] (17)
to the attacker. It responds to a ciphertext-type query and a challenge (H)IBE-key-type query in the same way as \(O_5\).

**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 \(I' \in \mathbb{Z}_m\), it chooses \(t \in \mathbb{Z}_n\) randomly and returns the group elements
\[(w_t^g h, v' g'' t, g' \cdot g_2, (u'_{\mu} h_0)' \cdot g_2)\] (18)
to the attacker. Upon receiving a ciphertext-type query for \(T' \in \mathbb{Z}_n\), it chooses \(t_0 \in \mathbb{Z}_n\) randomly and returns the group elements
\[(w_0^g h, v_0 g'' t_0, g'' h, (u''_{\mu} h_0)' \cdot g_2)\] (19)
to the attacker. It responds to a decryption-type query and a challenge (H)IBE-key-type query in the same way as \(O_6\).

**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 \(I, T \in \mathbb{Z}_m\) by choosing \(r, y', r', y'' \in \mathbb{Z}_n\) randomly and returns the group elements
\[(w' (g, g_3)^{y'^*}, g' (g, g_3)^{v'^*} (u'h)' \cdot g'), w''_0 (g, g_3)^{y''*}, g'' (g, g_3)^{v''*} (u''_{\mu} h_0)' \cdot g')\]
to the attacker. It responds to a challenge (H)IBE-key-type query in the same way as \(O_7\).

We define the advantage of an attacker \(A\) in distinguishing between \(O_0\) and \(O_7\) to be \(| Pr[A(O_0) = 1] - Pr[A(O_7) = 1] ||. \) Here, we assume that \(A\) interacts with either \(O_0\) or \(O_7\), 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-functional private key(or semi-functional update key) one by one since the adversary of RHIBE can query a private key for $ID|k \in \text{Prefix}(ID')$ 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 $\{SK_{HIBE,S_j}\}_{S_j \in \text{Path}}$ and an update key $UK_{ID|k-1,T,R}$ consists a randomized decryption key $RSK_{HIBE}$ and many IBE private keys $\{SK_{IBE,S_i}\}_{S_i \in \text{CVR}}$ 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 from 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 $L_j = \text{Label}(v_j)$. 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 $B$ answers the secret (update, and decryption) key queries of the adversary $A$ by requesting the associated (H)IBE private keys from an oracle simulator $O$, shown in Fig 1. When the adversary $A$ queries $B$ 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 $O$. $O$ adaptively answers $B$ the corresponding group elements which it constructs by using the public parameters given by some complexity assumption. Therefore, under the complexity assumptions, the oracle $O_i$ that $O$ chooses to answer $B$ is indistinguishable and consequently the adversary $A$ cannot distinguish whether $A$ is playing the real RHIBE game or other variation games based on all the answers $A$ recieves after the adaptive queries to $B$.

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 $B$ 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 \(v_j \in BT_{ID_{h1}}\), 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 \(v_j \in BT_{ID_{h1}}\) 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 \(ID'\) 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 \(Game_{ESF}\), the adversary types are formally defined as follows:

1. **Type-1**: An adversary is Type-1 if it queries on a hierarchical identity \(ID'_l \notin \text{Prefix}(ID')\) 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 \(ID'_l \in \text{Prefix}(ID'|l)\) for at least one HIBE private key with \(h\), or it may query on a hierarchical identity \(ID'_l \notin \text{Prefix}(ID')\) 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 sequence of games is basically 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 \(G_{E-S}\) 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 \(G_{E-U}\) 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

\[
|\text{Adv}_{A}^{G_{C}}(\lambda) - \text{Adv}_{A}^{G_{ESF}}(\lambda)| \leq O(q_s \log N_{max}) + O(q_s r_{max} \log N_{max})(\text{Adv}_{B}^{A3} + \text{Adv}_{B}^{A4}) + O(1)(\text{Adv}_{B}^{A3} + \text{Adv}_{B}^{A4})
\]

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 \(G_{ESF}\) 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.
### 4.4.2 Type-1 adversary.

As defined before, the Type-1 adversary is restricted to queries on a hierarchical identity ID_j = ID_l Prefix(ID_k). By querying 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, 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

| Oracle | CT-Type Response | SK-Type Response | UK-Type Response | DK-Type Response |
|--------|------------------|------------------|------------------|------------------|
| O_0    | SF               | Normal           | Normal           | Normal           |
| O_{1/2} | SF               | ESF-1            | Normal           | Normal           |
| O_1    | SF               | ESF-2            | Normal           | Normal           |
| O_r    | ESF-2'           | ESF-2            | Normal           | Normal           |
| O_{r'} | ESF-3'           | ESF-2            | Normal           | Normal           |
| O_{r''} | ESF-4'          | ESF-2            | Normal           | Normal           |
| O_2    | ESF-1            | ESF-2            | Normal           | Normal           |
| O_{v/2} | ESF-1            | ESF-2            | ESF-1            | Normal           |
| O_3    | ESF-1            | ESF-3            | ESF-2            | Normal           |
| O_3,1  | ESF-1            | ESF-3            | Normal           | Normal           |
| O_3,2  | ESF-1            | ESF-3            | Normal           | 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            | Normal           | Normal           |
| O_{3,6} | ESF-1            | ESF-4            | ESF-4            | Normal           |
| O_{v/2} | ESF-1            | ESF-4            | ESF-5            | Normal           |
| O_r    | ESF-2'           | ESF-4            | SF               | Normal           |
| O_{r'} | ESF-3'           | ESF-4            | SF               | Normal           |
| O_{r''} | ESF-4'          | ESF-4            | SF               | Normal           |
| O_{v/2} | SF               | ESF-5            | SF               | Normal           |
| O_4    | SF               | SF               | SF               | Normal           |

Note: oracles marked with † initialize with an extra G_p term on g^g-g.

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

#### Table 4. Simulation of challenge keys and ciphertext in oracles for the proof of the indistinguishability between GC and GSF under Type-1 adversary.

#### Table 5. Definition of games between G_ESF and G_SSF.

| Games | Oracles | Keys in Queries | Challenge Ciphertext |
|-------|---------|-----------------|----------------------|
| G_ESF | O_1     | ESF – 2         | Normal               |
| G_ESF | O_{x,1} | ESF – 3         | Normal               |
| G_ESF | O_{x,2} | ESF – 3         | Normal               |
| G_ESF | O_{x,3} | ESF – 3         | Normal               |
| G_ESF | O_{x,4} | ESF – 4         | Normal               |
| G_ESF | O_{x,5} | ESF – 4         | Normal               |
| G_ESF | O_{x,6} | ESF – 4         | Normal               |
| G_ESF | O_{x,7} | ESF – 4         | Normal               |
| G_ESF | O_{x,8} | ESF – 4         | Normal               |
| G_ESF | O_4     | SF              | SF                   |

https://doi.org/10.1371/journal.pone.0195204.t005
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 following Lemma 1, to Lemma 20, we obtain the advantage of Type-1 adversary to distinguish between $G_{C_0}$ and $G_{SF_0}$ as

$$Adv_{A}^{G_{C_0}} - Adv_{A}^{G_{SF_0}} \leq \left| Adv_{A}^{G_{C_0}} - Adv_{A}^{G_{SF_0}} \right| + \left| Adv_{A}^{G_{C_0}} - Adv_{A}^{G_{SF_0}} \right|$$

$$+ \left| Adv_{A}^{G_{C_0}} - Adv_{A}^{G_{SF_0}} \right| + \left| Adv_{A}^{G_{C_0}} - Adv_{A}^{G_{SF_0}} \right|$$

$$\leq (O(q_n(q_i + q_n)) + O(l))(Adv_{A}^{G_{C_0}} + Adv_{A}^{G_{SF_0}})$$

$$\leq (O(q_n \log N_{\max} + q_n r_{\max} \log N_{\max}) + O(l))(Adv_{A}^{G_{C_0}} + Adv_{A}^{G_{SF_0}})$$

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

**1. Indistinguishability of $G_{C_0}$ and $G_{SF_0}$**

For the security proof of the indistinguishability of $G_{C_0}$ and $G_{SF_0}$, we define a sequence of additional hybrid games $G_{C_0,1}, \ldots , G_{C_0,h}, \ldots , G_{C_0,n}$, where $G_{C_0} = G_{C_0,0}$ 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 $G_{C_0,h}$ 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 $O_{0,1}$ and provides the attacker with initial group elements from the same distribution. Upon receiving a challenge HIBE-key-type

| Oracle | CT-Type Response | SK-Type Response | UK-Type Response | DK-Type Response |
|--------|------------------|------------------|------------------|------------------|
| $O_0$  | SF               | Normal           | Normal           | Normal           |
| $O_{1/2}$ | SF               | Normal           | ESF-1            | Normal           |
| $O_1$  | SF               | Normal           | ESF-2            | Normal           |
| $O'_{1/2}$ | ESF-2'         | Normal           | ESF-2            | Normal           |
| $O_2$  | ESF-1            | Normal           | ESF-2            | Normal           |
| $O_{3/2}$ | ESF-1            | ESF-2            | ESF-2            | Normal           |
| $O_3$  | ESF-1            | ESF-2            | ESF-3            | Normal           |
| $O_{4,1}$ | ESF-1            | ESF-2            | ESF-3            | Normal           |
| $O_{4,2}$ | ESF-1            | ESF-2            | ESF-3            | Normal           |
| $O_{5,3}$ | ESF-3            | ESF-3            | ESF-3            | Normal           |
| $O_{5,4}$ | ESF-3            | ESF-3            | ESF-4            | Normal           |
| $O_{5,5}$ | ESF-3            | ESF-4            | ESF-4            | Normal           |
| $O_{6,6}$ | ESF-4            | ESF-4            | ESF-4            | Normal           |
| $O_{7,7}$ | ESF-5            | ESF-5            | ESF-5            | Normal           |
| $\tilde{O}_{1/2}$ | ESF-2'         | SF               | ESF-4            | Normal           |
| $\tilde{O}_{2}$ | ESF-3'         | SF               | ESF-4            | Normal           |
| $\tilde{O}_{3}^\ast$ | ESF-4'         | SF               | ESF-4            | Normal           |
| $O_{1/2}$  | SF               | SF               | ESF-5            | Normal           |
| $O_4$   | SF               | SF               | Normal           | Normal           |

Note: oracles marked with † initialize with an extra $G_p$, term on $g^g g$.

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

Table 6. Simulation of challenge keys and ciphertext in oracles under Type-2 adversary for the proof of the indistinguishability between $G_{C_0}$ and $G_{SF_0}$. 
query for $i \in \mathbb{Z}_n$, it chooses $r', y' \in \mathbb{Z}_n$ randomly, and also chooses $X_3, Y_3 \in G_{p_2}$ randomly. It returns the group elements

$$(w', g'^r, v'^r(u'h')^rX_3, g'^{y'}Y_3)$$

(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 $H_{1,1}, H_{1,2}, \ldots, H_{h,1}, H_{h,2}, \ldots, H_{q_1,1}, H_{q_1,2}$ where $H_{0,2} = G_{C,h}$ and $H_{q_1,2} = G_{C,h+1}$, and $q_1$ is the maximum number of HIBE private key queries for the node index $h$. The games are formally defined as follows:

**Game** $H_{h,1}$ This game $H_{h,1}$ for $1 \leq h \leq q_1$ is almost the same as $G_{z,h-1}$ except the generation of HIBE private keys and IBE private keys with the node $\theta_h$ of the index $h$. An IBE private key with an index pair $(h, i)$ is generated as normal. An HIBE private key with an index pair $(h, i)$ is generated as follows:

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

**Game** $H_{h,2}$ This game $H_{h,2}$ for $1 \leq h \leq q_1$ is almost the same as $H_{h,1}$ except the generation of HIBE private key with an index pair $(h, i)$ and $i = h$ is generated as a ESF-2 $SK_{HIBE,h}$ by using the element groups in Eq 22.

**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 $H_{h,1}$ and $H_{h,2}$ 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 $H_{h,1}$ and $H_{h,2}$ with non-negligible advantage.

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

$$Adv_A^{G_{C,h-1}} - Adv_A^{G_{C,h}} \leq \sum_{h=1}^{q_1} \left( |Adv_A^{H_{h,1}} - Adv_A^{H_{h,2}}| + |Adv_A^{H_{h,1}} - Adv_A^{H_{h,2}}| \right)$$

$$\leq O(q_1)(Adv_A^{G_{C}} + Adv_A^{A_{1/2}})$$

So we obtain the following equation

$$Adv_A^{G_{C}} - Adv_A^{G_{E,S}} \leq \sum_{h=1}^{q_1} (Adv_A^{G_{C,h-1}} - Adv_A^{G_{C,h}}) \leq O(q_1q_2)(Adv_A^{A_{1/2}} + Adv_A^{A_{1/2}})$$

(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 $q_k$ denote the number of ciphertext-type queries made by a PPT attacker $A$. Firstly we define hybrid games $S_{-1,1}, S_{0,2}, S_{0,3}, S_{0,1}, S_{1,2}, S_{1,3}, S_{1,1}, \ldots, S_{k,2}, S_{k,3}, S_{k,1}, \ldots, S_{k-1,2}, S_{k-1,3}, S_{k-1,1}$, where $S_{-1,1} = G_{E,S}$ and $S_{-1,1} = G_{E'-S}$. The games are formally defined as follows:

**Game** $S_{k,1}$ This game $S_{k,1}$ for $0 \leq k \leq q_1$ is almost the same as $G_{E,S}$ except the generation of the challenge ciphertext. The challenge ciphertext of $(T^r, I_1^r, \ldots, I_{q_1}^r)$ is generated as $EST^{2^k}$-CT outputted by $Encrypt_{ESF-2^k}$ defined in Appendix A.
Game $S_{k,2}$ This game $S_{k,2}$ for $0 \leq k \leq q_e - 1$ is almost the same as $G_{E-S}$ except the generation of the challenge ciphertext. The challenge ciphertext of $(T^*, I'_1, \ldots, I'_{q_e-1})$ is generated as $EST^k$-CT outputed by $EncrypESF^k$ defined in AppendixA.

Game $S_{k,3}$ This game $S_{k,3}$ for $0 \leq k \leq q_e - 1$ is almost the same as $G_{E-S}$ except the generation of the challenge ciphertext. The challenge ciphertext of $(T^*, I'_1, \ldots, I'_{q_e-1})$ is generated as $EST^k$-CT outputed by $EncrypESF^k$ defined in AppendixA.

We will define additional oracles $O'_i$ for each $i$ from 0 to $q_e - 1$, $O'_j$ for each $j$ from 0 to $q_e - 1$, and $O''_j$ for each $j$ from 0 to $q_e - 1$, to sample various distributions of group elements used for constructing the various types of ciphertexts in Game $S_{k,2}$, Game $S_{k,3}$ 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 $j$th 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, $O'_i$ is identical to $O_1$ and $O'_j$ is identical to $O_2$.

Oracle $O''_j$ 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 $I'_i$, it chooses a random $t \in Z_N$ and random elements $X_3, Y_3 \in G_{p_1}$ and responds with:

$$ (w_i g_2^{v_i} v_i^{X_3^{i}}, g^{r_i} X_3, (u_i h_i)^{Y_3}) $$(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 $X'_3, Y'_3 \in G_{p_1}$ and responds with:

$$ (w_0 g_2^{v_0} v_0^{X_3^{i}}, g^{r_0} X_3, (u_0 h_0)^{Y_3}) $$ (25)

Oracle $O''_j$ 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 $I'_i$, it chooses a random $t \in Z_N$ and random elements $X_3, Y_3 \in G_{p_1}$ and responds with:

$$ (w_i g_2^{v_i} v_i^{X_3^{i}}, g^{r_i} X_3, (u_i h_i)^{Y_3}) $$ (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 $X'_3, Y'_3 \in G_{p_1}$ and responds with:

$$ (w_0 g_2^{v_0} v_0^{X_3^{i}}, g^{r_0} X_3, (u_0 h_0)^{Y_3}) $$ (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 $S_{k,2}$ and $S_{k,3}$ with non-negligible advantage.
The games are formally defined as follows:

\[ \text{Adv}^S_{A, E} - \text{Adv}^S_{A, S} \leq \sum_{k=0}^{q-1} \left( |\text{Adv}^S_{A, k} - \text{Adv}^S_{A} | + |\text{Adv}^S_{A} - \text{Adv}^S_{A} | \right) \]

(28)

(3) Indistinguishability of \( G_{ES'} \) and \( G_{ESP} \)

For the security proof of the indistinguishability of \( G_{ES'} \) and \( G_{ESP} \), we define a sequence of additional hybrid games \( G_{S,1}, \ldots, G_{S,n}, \ldots, G_{S,q}, \ldots \) where \( G_{ES'} = G_{S,0} \) 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 \( G_{S,n} \), 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 \( O_{3/2} \). 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 \( i', y' \in Z_n \) randomly, and also chooses \( X_0, Y_3 \in G_{p_1} \) randomly. It returns the group elements

\[ (w_1', g', w_0' (u_0'h_0)' X_0, g' Y_3) \]

(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 \( E_{1,1}, E_{1,2}, \ldots, E_{h,1}, H_{h,2}, \ldots, E_{q,1}, E_{q,2} \) where \( E_{0,2} = G_{S,h} \) and \( E_{q,2} = G_{S,q+1} \), and \( q_e \) is the maximum number of IBE private key queries for the node index \( h \).

The games are formally defined as follows:

**Game** \( E_{h,1} \). This game \( E_{h,1} \) for \( 1 \leq h \leq q_e \) 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 \((h, i)\) is generated as ESF-2. An IBE private key with an index pair \((h, i)\) is generated as follows:

1. \( i < h \); It generates a normal \( SK'_{IBE,h} \) and converts the key to a ESF-2 \( SK_{IBE,h} \).
2. \( i = h \); 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. \( i > h \); It simply generates a normal IBE private key.

**Game** \( E_{h,2} \). This game \( E_{h,2} \) for \( 1 \leq h \leq q_e \) is almost the same as \( E_{h,1} \) except the generation of IBE private key with an index pair \((h, i)\) and \( i = h \) 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 \( O_{3/2} \) with non-negligible advantage. So no PPT attacker can distinguish between \( E_{h,-1,2} \) and \( E_{h,1} \) with non-negligible advantage.

**Lemma 7** Under Assumptions 4, no PPT attacker can distinguish between \( O_{3/2} \) and \( O_3 \) with non-negligible advantage. So no PPT attacker can distinguish between \( E_{h,1} \) and \( E_{h,2} \) with non-negligible advantage.
Let $\text{Adv}^G_A$ be the advantage of $A$ in a game $G_{E,B}$. From the Lemma 6, 7, we obtain the following equation

$$\text{Adv}^{G_{n-1},b}_{A} - \text{Adv}^{G_{n-1},b}_{A} \leq \sum_{k=i}^{b} (|\text{Adv}^{F_{h,1}}_{A} - \text{Adv}^{F_{h,2}}_{A}| + |\text{Adv}^{F_{h-1,2}}_{A} - \text{Adv}^{F_{h-1,1}}_{A}|)$$

$$\leq O(q) (\text{Adv}^{G^{1}}_{B} + \text{Adv}^{G^{1}}_{B})$$

So we obtain the following equation

$$\text{Adv}^{G_{n-1},b}_{A} - \text{Adv}^{G_{n-1},b}_{A} \leq \sum_{k=i}^{b} (|\text{Adv}^{G_{n-1},b}_{A} - \text{Adv}^{G_{n-1},b}_{A}|) \leq O(q) (\text{Adv}^{G^{1}}_{B} + \text{Adv}^{G^{1}}_{B})$$  (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 ciphertext in every game, and the decryption situation according to the types of keys and ciphertexts.

- $G_{ESF-1}$: 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.
- $G_{ESF-2}$: 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.
- $G_{ESF-3}$: The challenge ciphertext is changed to ESF-5. The secret keys are ESF-2. The update keys are still ESF-2. The decryption keys are still normal.
- $G_{ESF-4}$: The secret keys are changed to ESF-4. The update keys are still ESF-3. The challenge ciphertext is still ESF-4. The decryption keys are still normal.
- $G_{ESF-5}$: The challenge ciphertext is changed to ESF-4. The secret keys are ESF-4. The update keys are ESF-3. The decryption keys are still normal.
- $G_{ESF-6}$: The challenge ciphertext is changed to ESF-3. The secret keys are ESF-3. The update keys are ESF-2. The decryption keys are still normal.
- $G_{ESF-7}$: The challenge ciphertext is changed to semi-functional. The secret keys are ESF-4. The update keys are semi-functional. The decryption keys are still normal.
- $G_{ESF-8}$: The challenge ciphertext is changed to ESF-1. The secret keys are ESF-3. The update keys are ESF-3. The decryption keys are still normal.

We first 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 $G_{F,1}, G_{F,2}, \cdots, G_{F,2}$ 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 $G_{F,1}$, 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 $i_{p} \leq h$ are of ESF-3, the remaining HIBE private keys with a node index $i_{p} > h$ are ESF-2.

Oracle $O_{A,1}$ This oracle initializes in the same way as $O_{A}$ and provides the attacker with initial group elements from the same distribution. Upon receiving a challenge HIBE-key-type query for $i \in Z_{p}$, it chooses $r, r' \in Z_{p}$ randomly, and also chooses $X_{2}, Y_{2} \in G_{F,2}$ and $X_{3}, Y_{3} \in G_{p,2}$ randomly. It returns the group elements

$$\left( (w^{h} g^{i'}, g^{i'} g^{i'}, v^{h}) X_{2} X_{3}, g^{i'} Y_{2} Y_{3} \right)$$  (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 $F_1, \cdots, F_{h}$ where $F_0 = G_{S,h}$ and $F_{q_i} = G_{S,h+1}$, and $q_i$ is the maximum number of HIBE private key queries for the node index $h$. The games are formally defined as follows:

**Game $F_h$.** This game $F_h$ for $1 \leq h \leq q_i$ 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 $(h, i)$ is generated as ESF-2. An HIBE private key with an index pair $(h, i)$ 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 $F_{h-1}$ and $F_{h}$ with non-negligible advantage.

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

$$Adv_{A}^{G_{F,h}} - Adv_{A}^{G_{F,h-1}} \leq \sum_{h=1}^{q_i} (Adv_{A}^{G_{F,h-1}} - Adv_{A}^{G_{F,h}}) \leq O(q_{i}q_{j})$$(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 $G_{F,1}, \ldots, G_{F,1,0} \ldots, G_{F,1,q_h}$, where $G_{ESF-1} = G_{F,1,0}$ and $q_h$ 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 $1 \leq h \leq q_h$, 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 $r, y, y' \in Z_n$ 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

$$(w_0', g_1^r, g_3^y, g_2^{y'}, v_0'(u_0^r h_0), X_2X_3, g_2Y_2Y_3)$$ (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 $F_{1,1}, \ldots, F_{h,1}, \ldots, F_{q_h,j}$ where $F_{1,0} = G_{F,1,0}$ and $F_{q_h,j} = G_{F,1,h+1}$, and $q_h$ is the maximum number of IBE private key queries for the node index $h$. The games are formally defined as follows:

**Game $F_{1,h}$.** This game $F_{1,h}$ for $1 \leq h \leq q_h$ 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 $(h, i)$ is generated as ESF-3. An IBE private key with an index pair $(h, i)$ is generated as follows:

1. $i_c \leq h_c$: It generates a ESF-3 $SK_{IBE,h}$
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 $F_{h-1}$ and $F_{h}$ with non-negligible advantage.
Let $\text{Adv}_{A}^{G_{F^{1.-1}}} \in G_{F^{1.-1}}$. From the Lemma 9, we obtain the following equation

$\text{Adv}_{A}^{G_{F^{1.-1}}} - \text{Adv}_{A}^{G_{F^{1.-1}}} \leq \sum_{h=1}^{q_h} (\text{Adv}_{A}^{G_{F^{1.-1}}} - \text{Adv}_{A}^{G_{F^{1.-1}}}) \leq O(q_h, q_z) (\text{Adv}_{A}^{G^{3}})$ (34)

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

**Oracle $O_{3,3}$:** This oracle initializes a bit differently from the other oracles. It fixes random elements $g, u, h, v, w, u_0, h_0, v_0, w_0 \in G_{F}, g_2 \in G_{F}, g_3 \in G_{F}$. It chooses random exponents $s, \gamma, \beta, \delta, y, \psi, \sigma, \sigma', \alpha', b', t_1, t_2, t'_3 \in Z_N$. It initially provides the attacker with the group elements:

$$ (g, u, h, v, w, g' (g_2 g_3)^{w}, w'^{(g_2 g_3)^{w}}, g'^{(g_2 g_3)^{w}}, v'^{(g_2 g_3)^{w}}, u_0, h_0, v_0, w_0, g'^{(g_3 g_3)^{w}}, v'^{(g_2 g_3)^{w}}) $$ (35)

What differs from the previous oracles here is the added $g_3$ and term: notice that this is uniformly random in $G_{F}$, since $\gamma$ is random modulo $p_2$ (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:

$$ (w'^{(g_2 g_3)^{w}})^t (g_2 g_3)^t, g'^{(g_2 g_3)^{w}}, v'^{(g_2 g_3)^{w}} ) $$ (36)

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

$$ (w'^{(g_2 g_3)^{w}})^t (g_2 g_3)^t, g'^{(g_2 g_3)^{w}}, v'^{(g_2 g_3)^{w}} ) $$ (37)

It is crucial to note that these $G_{F}$ terms are the 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 $G_{F^{3}-2}$ and $G_{F^{3}-3}$ with non-negligible advantage.

From the Lemma 10, we obtain the following equation

$$ \text{Adv}_{A}^{G_{F^{3}-2}} - \text{Adv}_{A}^{G_{F^{3}-3}} \leq (\text{Adv}_{A}^{G^{3}}) $$ (38)

**Indistinguishability of $G_{F^{3}-3}$ and $G_{F^{3}-4}$:** For the security proof of the indistinguishability of $G_{F^{3}-3}$ and $G_{F^{3}-4}$, we define a sequence of games additional hybrid games $G_{F^{3,1}}$, $G_{F^{3,2}}, \ldots, G_{F^{3,h}}$, where $G_{F^{3,1}} = G_{F^{3,0}}$ and $q_h$ 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 $y, r \in Z_N$, $X_2, Y_2 \in G_{F}$, and $X_3, Y_3 \in G_{F}$, and responds with:

$$ (w'^{(g_2 g_3)^{w}}, g'^{(g_2 g_3)^{w}}, v'^{(u'h)^{(X_2 X_3, g'Y_2 Y_3)}} ) $$ (39)
We define hybrid games $F_3_1, \ldots, F_3_h, \ldots, F_3_q$, where $F_3_0 = G_{F-3,h}$ and $F_3_q = G_{F-3,h+1}$, and $q_e$ is the maximum number of HIBE private key queries for the node index $h$. The games are formally defined as follows:

**Game** $F_3_h$. This game $F_3_h$ for $1 \leq h \leq q_e$ 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. $i_c \leq h$: It generates a ESF-4 SK$_{HIBE,h}$ by using the element groups in Eq 39.
2. $i_c > h$: 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 $Adv^G_{F-3,h}$ be the advantage of $A$ in a game $G_{F-3,h}$. From the Lemma 11, we obtain the following equation

$$Adv^G_{ESF-3} - Adv^G_{ESF-4} \leq \sum_{h=1}^{q_e} (Adv^G_{ESF-3} - Adv^G_{ESF-4}) \leq O(q_e q_e)(Adv^G_{ESF})$$ (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 $G_{F-4,0}, \ldots, G_{F-4,q_n}$, 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 ESF-4, and 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 ciphertext 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:

$$(w_0^h (g_3 g_3)^r, g_3^r (g_3 g_3)^r, w_0^h (u_0^r h_0)^r X_2 X_3, g_3^r Y_2 Y_3)$$ (41)

We define hybrid games $F_4_1, \ldots, F_4_h, \ldots, F_4_q$, where $F_4_0 = G_{F-4,h}$ and $F_4_q = G_{F-4,h+1}$, and $q_e$ is the maximum number of IBE private key queries for the node index $h$. The games are formally defined as follows:

**Game** $F_4_h$. This game $F_4_h$ for $1 \leq h \leq q_e$ 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. $i_c \leq h$: It generates a ESF-4 SK$_{IBE,h}$ by using the element groups in Eq 41.
2. $i_c > h$: 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 \( \text{Adv}^{G_{F^{-1}}_{A}}_{A} \) be the advantage of \( A \) in a game \( G_{F^{-1},b} \). From the Lemma 12, we obtain the following equation

\[
\text{Adv}^{G_{ESF-5}}_{A} - \text{Adv}^{G_{ESF-5}}_{A} \leq \sum_{h=1}^{q_{b}} (\text{Adv}^{G_{F^{-1},b-1}}_{A} - \text{Adv}^{G_{F^{-1},b}}_{A}) \leq O(q_{n}q_{e}) (\text{Adv}^{A}_{A})
\]

### 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 \( g, u, h, v, w, u_{0}, v_{0}, w_{0} \in G_{F}, g_{2}, g_{3} \in G_{F} \). It chooses random exponents \( s, \gamma, \delta_{1}, \delta_{2}, y, y_{0}, \sigma_{1}, \sigma_{2}, d', b', t_{3}, t'_{3}, t_{3} \in Z_{N} \). It initially provides the attacker with the group elements:

\[
(g, u, h, v, w, u_{0}, v_{0}, w_{0}) \quad \text{and} \quad (g_{2}, g_{3})
\]

What differs from the previous oracles here is the added \( g_{3} \) and term: notice that this is uniformly random in \( G_{F} \) since \( \gamma \) is random modulo \( p_{3} \) (and uncorrelated from its value modulo \( p_{3} \)). 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 variables \( t \in Z_{N} \) and responds with:

\[
(w'_{0} g_{2}^{t} v_{0} g_{3}, g_{3}) = (u t_{3} h_{0})^{t} g_{2}^{t} g_{3}
\]

To answer a ciphertext-type query for \( T \), it chooses random variables \( t \in Z_{N} \) and responds with:

\[
(w'_{0} g_{2}^{t} v_{0} g_{3}, g_{3}) = (u t_{3} h_{0})^{t} g_{2}^{t} g_{3}
\]

It is crucial to note that these \( G_{F} \) terms are the 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

\[
\text{Adv}^{G_{ESF-5}}_{A} - \text{Adv}^{G_{ESF-6}}_{A} \leq (\text{Adv}^{A}_{A})
\]

### 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 \( G_{F^{-1},b}, \ldots, G_{F^{-1},b}, \ldots, G_{F^{-1},b} \) where \( G_{ESF-6} = G_{F^{-1},b} \) and \( q_{b} \) is the number of all node identifiers that are used in HIBE private keys and IBE private keys of an adversary. In the game \( G_{F^{-1},b} \) 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 \( i_{b} > h \) are ESF-4.

**Oracle** \( O_{3.12} \) This oracle initializes in the same way as \( O_{3,6} \) and provides the attacker with the 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 \( X_{2}, Y_{2} \in G_{F} \) and \( X_{3}, Y_{3} \in G_{F} \) randomly. It returns the group elements

\[
(w'_{0} g_{2}^{y} g_{3}^{y'}, g_{3}^{y'}, v_{0}^{y'} u_{0}^{y} h_{0} X_{2} Y_{3})
\]

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 $F_{6,1,1}$, $F_{6,1,2}$, $F_{6,2,1}$, $F_{6,2,2}$, $F_{6,q_e,1}$, $F_{6,q_e,2}$, where $F_{6,0,2} = G_{F^{6,0}}$ and $F_{6,q_e,2} = G_{F^{6,q_e+1}}$, and $q_e$ is the maximum number of IBE private key queries for the node index $h$. The games are formally defined as follows:

**Game** $F_{6,h,1}$ This game $F_{6,h,1}$ for $1 \leq h \leq q_e$ 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 $(h, i)$ is generated as ESF-4. An IBE private key with an index pair $(h, i)$ is generated as follows:

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

**Game** $F_{6,h,2}$ This game $F_{6,h,2}$ for $1 \leq h \leq q_e$ is almost the same as $G_{F^{6,h}}$, except the generation of IBE private key with an index pair $(h, i)$ and $i_c$ 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 $F_{6,h,1}$ and $F_{6,h,2}$ with non-negligible advantage.

**Oracle** $O^{+}$ This oracle initializes in the same way as $O^{+}$, and provides the attacker with initial group elements from the same distribution. It also responds to a ciphertext-type query as same as $O^{+}$. 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 $T \in Z_m$ it chooses $r, y' \in Z_m$ randomly, and returns the group elements

$$
(w_i' (g_{g_i} y')^r ; g_i' (g_{g_i} y')^r (u_i h_i)^r ; g_i' (g_{g_i} y')^r)
$$

(48)

to the attacker.

**Lemma 15** Under Assumptions 3, no PPT attacker can distinguish between $O_{7/2}$ and $O^{+}_e$ with non-negligible advantage. So no PPT attacker can distinguish between $F_{6,h,1}$ and $F_{6,h,2}$ with non-negligible advantage.

Let $Adv^G_{A}$ be the advantage of $A$ in a game $G_{F,h}$. From the Lemma 14, 15, we obtain the following equation

$$
Adv^G_{A} - Adv^G_{A} \leq \sum_{h=1}^{q_e} (|Adv^{F_{6,h,1}}_{A} - Adv^{F_{6,h,2}}_{A}| + |Adv^{F_{6,h-1,2}}_{A} - Adv^{F_{6,h-1}}_{A}|) \\
\leq O(q_e)(Adv^{K_{i}} + Adv^{K_{i}})
$$

So we obtain the following equation

$$
Adv^G_{A} - Adv^G_{A} \leq \sum_{h=1}^{q_e} (Adv^{G_{h-1}}_{A} - Adv^{G_{h-1}}_{A}) \leq O(q_e)(Adv^{K_{i}} + Adv^{K_{i}}) \quad (49)
$$

**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 $q_e$ denote the number of ciphertext-type queries made by a PPT attacker $A$. Firstly we define hybrid games $S_{1,1,1}^{'}, S_{1,2,1}^{'}, S_{2,1,1}^{'}, S_{2,2,1}^{'}, S_{3,1,1}^{'}, \ldots, S_{3,2,1}^{'}, S_{3,3,1}^{'}, \ldots, S_{1,2,1}^{'}, S_{1,3,1}^{'}, S_{1,3,1}^{'}, S_{2,1,1}^{'}, S_{2,2,1}^{'}, S_{2,3,1}^{'}, S_{3,1,1}^{'}, S_{3,2,1}^{'}, S_{3,3,1}^{'}, \ldots, S_{4,1,1}^{'}, S_{4,2,1}^{'}, S_{4,3,1}^{'}, S_{4,4,1}^{'}, S_{4,4,1}^{'}, S_{5,1,1}^{'}, S_{5,2,1}^{'}, S_{5,3,1}^{'}, S_{5,4,1}^{'}, \ldots, S_{6,1,1}^{'}, S_{6,2,1}^{'}, S_{6,3,1}^{'}, S_{6,4,1}^{'}, S_{6,5,1}^{'}, S_{6,6,1}^{'}, S_{7,1,1}^{'}, S_{7,2,1}^{'}, S_{7,3,1}^{'}, S_{7,4,1}^{'}, \ldots, S_{q_e,1,1}^{'}, \ldots, S_{q_e,2,1}^{'}, S_{q_e,3,1}^{'}, \ldots, S_{q_e,1,1}^{'}, \ldots, S_{q_e,1,1}^{'}, G_{ESF^{-7}}$ and $G_{ESF^{-8}}$. The games are formally defined as follows:
Game \( S_{k,1} \). This game \( S_{k,1} \) for \( 0 \leq k \leq q_e \) is almost the same as \( G_{ESP-7} \) except the generation of the challenge ciphertext. The challenge ciphertext of \( (T^*, I_1^*, \ldots, I_{q_e-1}^*) \) is generated as \( EST-2^k \)-CT outputed by Encrypt\( ESF-2^k \).

Game \( S_{k,2} \). This game \( S_{k,2} \) for \( 0 \leq k \leq q_e - 1 \) is almost the same as \( G_{ESP-7} \) except the generation of the challenge ciphertext. The challenge ciphertext of \( (T^*, I_1^*, \ldots, I_{q_e-1}^*) \) is generated as \( EST-3^k \)-CT outputed by Encrypt\( ESF-3^k \).

Game \( S_{k,3} \). This game \( S_{k,3} \) for \( 0 \leq k \leq q_e - 1 \) is almost the same as \( G_{ESP-7} \) except the generation of the challenge ciphertext. The challenge ciphertext of \( (T^*, I_1^*, \ldots, I_{q_e-1}^*) \) is generated as \( EST-4^k \)-CT outputed by Encrypt\( ESF-4^k \).

We will define additional oracles \( O_i^r \) for each \( i \) from 0 to \( q_e - 1 \), \( \tilde{O}_i^r \) for each \( i \) from 0 to \( q_e - 1 \), and \( \tilde{O}_i^n \) for each \( i \) from 0 to \( q_e - 1 \).

Oracle \( \tilde{O}_i^r \) This oracle acts the same as \( O_i^r \) except that its response to the ciphertext-type query is as same as \( O_i^r \).

Oracle \( \tilde{O}_i^n \) This oracle acts the same as \( O_i^r \) except that its response to the ciphertext-type query is as same as \( O_i^r \).

Lemma 16 Under Assumptions 3, no PPT attacker can distinguish between \( \tilde{O}_i^r \) and \( O_i^r \) with non-negligible advantage. So no PPT attacker can distinguish between \( S_{k,1} \) and \( S_{k,2} \) with non-negligible advantage.

Lemma 17 Under Assumptions 4, no PPT attacker can distinguish between \( \tilde{O}_i^r \) and \( O_i^r \) with non-negligible advantage. So no PPT attacker can distinguish between \( S_{k,1} \) and \( S_{k,2} \) with non-negligible advantage.

Lemma 18 Under Assumptions 3, no PPT attacker can distinguish between \( \tilde{O}_i^r \) and \( O_i^r \) with non-negligible advantage. So no PPT attacker can distinguish between \( S_{k,1} \) and \( S_{k,2} \) with non-negligible advantage.

Let \( Adv_{A_{S_{k,1}}}^S \), \( Adv_{A_{S_{k,2}}}^S \) and \( Adv_{A_{S_{k,3}}}^S \) be the advantage of \( A \) in the games \( S_{k,1} \), \( S_{k,2} \) and \( S_{k,3} \). From the Lemma 16, 17, 18, we obtain the following equation

\[
Adv_{A_{G_{ESP-7}^k}}^S - Adv_{A_{G_{ESP-8}^k}}^S \leq \sum_{i=0}^{q_e-1} (|Adv_{A_{S_{k,1}}}^S - Adv_{A_{S_{k,2}}}^S| + |Adv_{A_{S_{k,2}}}^S - Adv_{A_{S_{k,3}}}^S|)
\]

\[
+ |Adv_{A_{S_{k,3}}}^S - Adv_{A_{S_{k,1}}}^S|
\]

\[
\leq q_e(2Adv_{A_{G_{ESP-7}^k}}^S + Adv_{G_{ESP-8}^k})
\]

Indistinguishability of \( G_{ESP-8}^k \) and \( G_{ESP-8}^k \). For the security proof of the indistinguishability of \( G_{ESP-8}^k \) and \( G_{ESP-8}^k \), we define a sequence of games \( G_{ESP-8,1} \), \( G_{ESP-8,2} \), \ldots, \( G_{ESP-8,q_e} \) where \( G_{ESP-8,1} = G_{ESP-8,0} \) and \( q_e \) is the number of all node identifiers that are used in HIBE private keys and IBE private keys of an adversary. In the game \( G_{ESP-8,h} \) for \( 0 \leq h \leq q_e \), the challenge ciphertext is semi-functional, all IBE private keys are semi-functional, HIBE private keys with a node index \( i_s \leq h \) are semi-functional, the remaining HIBE private keys with a node index \( i_n > h \) are ESF-4.

Oracle \( O_{2/3} \) This oracle initializes in the same way as \( O_i^r \) and provides the attacker with initial group elements from the same distribution. Upon receiving a challenge HIBE-key-type query for \( i \in Z_{q_e} \) it chooses \( r, r' \in Z_{q_e} \) randomly, and also chooses \( X_2, X_3, Y_3 \in G_{p_2} \) and \( X_3, Y_3 \in G_{p_3} \) randomly. It returns the group elements

\[
(w^{r}(g_3 g_3')^{r'} g^{r'} (g_3 g_3')^{r'})^{v'} \cdot (u' h)' X_2 X_3, g' Y_1)
\]
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 $I_{1,1}, I_{1,2}, \cdots, I_{h_c,1}, I_{h_c,2}, \cdots, I_{q_c,1}, I_{q_c,2}$ where $I_{0,2} = G_{F^{r-q_c}}$ and $I_{q_c,2} = G_{F^{r-q_c-1}}$, and $q_c$ is the maximum 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 $1 \leq h_c \leq q_c$ is almost the same as $G_{F^{r-q_c}}$ 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. $i_c < h$: It generates a semi-functional $SK_{HIBE,h}$
2. $i_c = h$: It generates a ESF-5 $SK_{HIBE,h}$
3. $i_c > h$: It generates a ESF-4 $SK_{HIBE,h}$

**Game $H_{h_c}$** This game $H_{h_c}$ for $1 \leq h_c \leq q_c$ is almost the same as $H_{h_c}$ 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 $O'_5$ and $O_{y/2}$ 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 $O_{y/2}$ and $O_4$ with non-negligible advantage. So no PPT attacker can distinguish between $I_{h_{c-1},1}$ and $I_{h_{c-2},2}$ with non-negligible advantage.

Let $Adv^{G_{F^{r-h}}}_A$ be the advantage of $A$ in a game $G_{F^{r-h}}$. From the Lemma 19, 20, we obtain the following equations

$$Adv^{G_{F^{r-h-1}}}_A - Adv^{G_{F^{r-h}}}_A \leq \sum_{h_{c-1}}^{q_c} (\left|Adv^{h_{c-1}}_A - Adv^{h_{c-2}}_A\right| + \left|Adv^{h_{c-1,2}}_A - Adv^{h_{c-1,1}}_A\right|)$$

$$\leq O(q_c)(Adv^{A_3}_B + Adv^{A_4}_B)$$

and

$$Adv^{G_{F^{r-h-2}}}_A - Adv^{G_{F^{r-h}}} \leq \sum_{h_{c-1}}^{q_c} (Adv^{G_{F^{r-h-1}}}_A - Adv^{G_{F^{r-h-2}}}_A) \leq O(q_c,q_c)(Adv^{A_3}_B + Adv^{A_4}_B)$$

So we obtain the following equation

$$Adv^{G_{F^{r-h-1}}}_A - Adv^{G_{F^{r-h}}} \leq |(Adv^{G_{F^{r-h}}}_A - Adv^{G_{F^{r-h-1}}}_A) + \sum_{i=1}^{r} (Adv^{G_{F^{r-h}}}_A - Adv^{G_{F^{r-h+1}}}_A) + |(Adv^{G_{F^{r-h-1}}}_A - Adv^{G_{F^{r-h}}}_A)| \leq O(q_c,q_c)(Adv^{A_3}_B) + O(q_c,q_c)(Adv^{A_3}_B) + O(q_c,q_c)(Adv^{A_4}_B) +$$

$$O(q_c)(2Adv^{A_3}_B + Adv^{A_4}_B) + O(q_c,q_c)(Adv^{A_3}_B + Adv^{A_4}_B) \leq O(q_c, q_c + q_c) + O(l)(Adv^{A_3}_B + Adv^{A_4}_B)$$

(53)
According to the equations Eqs 23, 28, 30, 53, we obtain the following equation
\[
Adv^{G_{ESF}}_A - Adv^{G_{SF'}}_A \leq |(Adv^{G_{ESF}}_A - Adv^{G_{SF'}}_A) + |(Adv^{G_{ESF}}_A - Adv^{G_{SF'}}_A)| \\
+ |(Adv^{G_{ESF}}_A - Adv^{G_{SF'}}_A)| + |(Adv^{G_{SF'}}_A - Adv^{G_{SF'}}_A)| \\
\leq (O(q_{n}(q_{+}q_{j})) + O(l))(Adv^{G_{SF'}}_B + Adv^{G_{SF'}}_B) \\
\leq (O(q_{n}\log N_{max}) + O(q_{n}r_{max}\log N_{max}) + O(l))(Adv^{G_{SF'}}_B + Adv^{G_{SF'}}_B)
\]

(54)

4.4.3 Type-2 adversary. The Type-2 adversary is restricted to queries on the update keys on the time \( T \subseteq T \). 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 \( G_{E_{K-D}} \) 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 swapped 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 \( G_{SF}, G_{E_{K-D}}, G_{ESF} \) 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 \((T, id_{1}, \cdots, id_{d})\) and follow the proof strategy in the nested dual system encryption of the unbounded HIBE [12].

We show the oracles for proving the indistinguishability of \( G_{SF} \) and \( G_{SF} \) in Table 7 which answer queries from the challenger \( B \) 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

\[
g, u, v, w, g^{u}, g^{v}, g^{w}, g^{u}, g^{v}, g^{w}, u^{v} g^{w}, v^{u} g^{w}, w^{u} g^{v}, w^{v} g^{u}, w^{u} g^{v}, w^{v} g^{u},\]

\[
u_{0}, v_{0}, w_{0}, u^{v} g^{w}, v^{u} g^{w}, w^{u} g^{v} g^{w}, v^{u} g^{w} g^{v}, w^{u} g^{v} g^{w}, v^{u} g^{w} g^{v},\]

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

(1) Indistinguishability of \( G_{SF} \) and \( G_{E_{K-D}} \)

For the security proof of the indistinguishability of \( G_{SF} \) and \( G_{E_{K-D}} \), we define a sequence of additional hybrid games \( J_{1,1}, J_{1,2}, \cdots, J_{h_{1,1}}, J_{h_{1,2}}, \cdots, J_{h_{1,2}}, J_{h_{2,2}} \) where \( J_{0,2} = G_{SF} \) and \( J_{h_{2,2}} = G_{E_{K-D}} \).
Table 7. Simulation of challenge keys and ciphertext 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 |
|----------|------------------|------------------|------------------|------------------|
| $O_4$    | SF               | SF               | SF               | Normal           |
| $O_{9/2}$| SF               | SF               | SF               | ESF-1            |
| $O_5$    | SF               | SF               | SF               | ESF-2            |
| $O_{9}$  | ESF-2'           | SF               | SF               | ESF-2            |
| $O_6$    | ESF-3'           | SF               | SF               | ESF-2            |
| $O_{9/2}$| ESF-4'           | SF               | SF               | ESF-2            |
| $O_{9}$  | ESF-1            | SF               | SF               | ESF-3            |
| $O_{9/2}$| ESF-3'           | SF               | SF               | ESF-3            |
| $O_6$    | SF               | SF               | SF               | ESF-4            |
| $O_{9/2}$| ESF-4'           | SF               | SF               | ESF-4            |
| $O_6$    | SF               | SF               | SF               | ESF-5            |
| $O_{9/2}$| SF               | SF               | SF               | SF               |

Note: oracles marked with † initialize with an extra $G_{p_1}$ term on $g'_1g$.  

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 $r$, $r'$, $r'' \in Z_n$ randomly, and also chooses $X_3$, $Y_3$, $X'_3$, $Y'_3 \in G_{p_2}$, $X_3$, $Y_3$, $X'_3$, $Y'_3 \in G_{p_3}$ randomly. It returns the group elements

\[(w', g', \nu' (u'h)X_3, g'Y_3, w_{0'}, g_{0'} \nu' (u'_h)X'_3, g'Y'_3)\]  \hspace{1cm} (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** $I_{h,d}^4$ This game $I_{h,d}^4$ for $1 \leq h_d \leq q_d$ is almost the same as $G_{SF'}$ except the generation of the decryption keys $DK_{ID_i,T}$.

1. $i_d < h_d$ It generates a normal $DK_{ID_i,T}^r$ and converts the key to a ESF-2 $DK_{ID_i,T}$ by using the element groups in Eq 17.
2. $i_d = h_d$ It generates a normal $DK_{ID_i,T}^r$ and converts the key to a ESF-1 $DK_{ID_i,T}$ by using the element groups in Eq 55.
3. $i_d > h_d$ It simply generates a normal decryption key.

**Game** $I_{h,d}^5$ This game $I_{h,d}^5$ for $1 \leq h_d \leq q_d$ is almost the same as $I_{h,d}^4$ except the generation of the $h_d$ decryption key is generated as a ESF-2 $DK_{ID_i,T}$ 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 $I_{h,d-1,2}$ and $I_{h,d}$ with non-negligible advantage.
Lemma 22 Under Assumptions 4, no PPT attacker can distinguish between \( \mathcal{O}_{0/2} \) and \( \mathcal{O}_{2} \) with non-negligible advantage. So no PPT attacker can distinguish between \( \mathcal{I}_{h_{n,1}} \) and \( \mathcal{I}_{h_{n,2}} \) with non-negligible advantage.

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

\[
\text{Adv}_{A}^{\text{Game}^{\text{ESF}}_{\mathcal{O}}^n} - \text{Adv}_{A}^{\text{Game}^{\text{ESF}}_{\mathcal{O}}^{n-1}} \leq \sum_{i=1}^{q_{L}} (|\text{Adv}_{A}^{\mathcal{O}_{i,1}} - \text{Adv}_{A}^{\mathcal{O}_{i,2}}| + |\text{Adv}_{A}^{\mathcal{O}_{i,1}} - \text{Adv}_{A}^{\mathcal{O}_{i,2}}|) \\
\leq O(q_{L})(\text{Adv}_{\mathcal{O}}^{\mathcal{E}_{5}} + \text{Adv}_{\mathcal{O}}^{\mathcal{E}_{6}})
\]  

(2) Indistinguishability of \( \mathcal{G}_{E-D} \) and \( \mathcal{G}_{\mathcal{E}} \)

We now prove the indistinguishability of \( \mathcal{G}_{E-D} \) and \( \mathcal{G}_{\mathcal{E}} \) in a hybrid argument using polynomially many steps. We let \( q_{L} \) denote the number of ciphertext-type queries made by a PPT attacker \( \mathcal{A} \). Firstly we define hybrid games \( L_{-1,1}, L_{0,1}, L_{0,3}, L_{0,1,1}, L_{1,1,1}, L_{1,2}, L_{1,3}, L_{2,1,1}, L_{2,2,2}, L_{2,3,3}, L_{3,1,1}, L_{3,2,2}, L_{3,3,3}, L_{4,1,1}, L_{4,2,2}, L_{4,3,3}, L_{5,1,1}, L_{5,2,2}, L_{5,3,3}, L_{6,1,1}, L_{6,2,2}, L_{6,3,3} \), where \( L_{-1,1} = \mathcal{G}_{E-D} \) and \( L_{q_{L}-1,1} = \mathcal{G}_{\mathcal{E}} \). The games are formally defined as follows:

**Game** \( L_{k,1} \) This game \( L_{k,1} \) for \( 0 \leq k \leq q_{L} \) is almost the same as \( \mathcal{G}_{E-D} \) except the generation of the challenge ciphertext. The challenge ciphertext of \( (T^{*}, I_{1}', \ldots, I_{q_{L}-1}') \) is generated as \( \text{EST-2}^{5} \)-CT outputed by \( \text{EncryptESF}^{3} \).

**Game** \( L_{k,2} \) This game \( L_{k,2} \) for \( 0 \leq k \leq q_{L} - 1 \) is almost the same as \( \mathcal{G}_{E-D} \) except the generation of the challenge ciphertext. The challenge ciphertext of \( (T^{*}, I_{1}', \ldots, I_{q_{L}-3}') \) is generated as \( \text{EST-3}^{5} \)-CT outputed by \( \text{EncryptESF}^{3} \).

**Game** \( L_{k,3} \) This game \( L_{k,3} \) for \( 0 \leq k \leq q_{L} - 1 \) is almost the same as \( \mathcal{G}_{E-D} \) except the generation of the challenge ciphertext. The challenge ciphertext of \( (T^{*}, I_{1}', \ldots, I_{q_{L}-2}') \) is generated as \( \text{EST-4}^{5} \)-CT outputed by \( \text{EncryptESF}^{4} \).

We will define additional oracles \( \hat{O}_{i}^{*} \) for each \( i \) from 0 to \( q_{L} - 1 \), \( \hat{O}_{i}^{\prime} \) for each \( i \) from 0 to \( q_{L} - 1 \), and \( \hat{O}_{i}^{\prime \prime} \) for each \( i \) from 0 to \( q_{L} - 1 \).

**Oracle** \( \hat{O}_{i}^{*} \) This oracle initializes in the same way as \( \mathcal{O}_{0}, \mathcal{O}_{6} \), and \( \mathcal{O}_{i} \) provides the attacker with initial group elements from the same distribution. It also responds to challenge key-type queries in the same way as \( \mathcal{O}_{0}, \mathcal{O}_{6} \). 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 \( i^{th} \) ciphertext-type query for some \( I_{i} \), if \( j \leq i \), it responds exactly like \( \mathcal{O}_{0} \); if \( j > i \), it responds exactly like \( \mathcal{O}_{6} \); in particular, \( \hat{O}_{0}^{*} \) is identical to \( \mathcal{O}_{0} \) and \( \hat{O}_{q_{L}}^{*} \) is identical to \( \mathcal{O}_{6} \).

**Oracle** \( \hat{O}_{i}^{\prime} \) This oracle acts the same as \( \hat{O}_{i}^{*} \) except in its response to the \( i^{th} \) ciphertext-type query. For the \( i^{th} \) ciphertext-type query for identity \( I_{i} \), it chooses a random \( t \in \mathbb{Z}_{N} \) and random elements \( X_{3}, Y_{3} \in G_{p_{i}} \) and responds with:

\[
(w_{i}^{\hat{O}_{i}^{*}}, vX_{3}^{i}, g^{X_{3}^{i}}, (u_{i}^{t} h)^{i} Y_{3})
\]

(57)

If \( i = 0 \), the \( i^{th} \) ciphertext-type query is for time \( T^{*} \). It chooses a random \( t_{0} \in \mathbb{Z}_{N} \) and random elements \( X_{3}, Y_{3} \in G_{p_{i}} \) and responds with:

\[
(w_{i}^{\hat{O}_{i}^{*}}, v_{0}X_{3}, g^{X_{3}}(u_{0}^{t_{0} h_{0}})^{i} Y_{3})
\]

(58)

**Oracle** \( \hat{O}_{i}^{\prime \prime} \) This oracle acts the same as \( \hat{O}_{i}^{*} \) except in its response to the \( i^{th} \) ciphertext-type query. For the \( i^{th} \) ciphertext-type query for identity \( I_{i} \), it chooses a random \( t \in \mathbb{Z}_{N} \) and random elements \( X_{3}, Y_{3} \in G_{p_{i}} \) and responds with:

\[
(w_{i}^{\hat{O}_{i}^{*}}, vX_{3}^{i}, g^{X_{3}^{i}}(u_{i}^{t} h)^{i} g_{2}^{i(d_{i}^{T} + b)} Y_{3})
\]

(59)

If \( i = 0 \), the \( i^{th} \) ciphertext-type query is for time \( T^{*} \). It chooses a random \( t_{0} \in \mathbb{Z}_{N} \) and random
elements $X_i', Y_i' \in G_{F_1}'$ and responds with:

$$
(w'_0 g_1^{h_0} v'_0 X'_i \sigma_2, g^6_2 g_2^{h_0} X'_i, (u_0^7 h_0)^{14} g_2^k (d_{h_0} + k) Y'_i))
$$

(60)

**Lemma 23** Under Assumptions 3, no PPT attacker can distinguish between $\tilde{O}_{i-1}$ and $\tilde{O}_{i}$ 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 $\tilde{O}_{i}$ and $\tilde{O}_{i}'$ 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 $\tilde{O}_{i}$ and $\tilde{O}_{i}'$ 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

$$
Adv^{G_{ESF}}_{\tilde{A}} - Adv^{G_{SF}}_{\tilde{A}} \leq \sum_{k=0}^{k-1} (|Adv^{G_{ESF}}_{\tilde{A}}| - Adv^{G_{SF}}_{\tilde{A}}) + (Adv^{G_{ESF}}_{\tilde{A}} - Adv^{G_{SF}}_{\tilde{A}})
$$

$$
\leq O(q_i)(2Adv^{A_{E}}_{B} + Adv^{A_{F}}_{B})
$$

(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 ciphertext 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. Definition of games between $G_{ESF}$ and $G_{SF}$.

| Games | Oracles | Keys in Queries | Challenge Ciphertext |
|-------|---------|-----------------|----------------------|
|       |         | SK              | UK               | DK              | Normal | SF | ESF – 2 | ESF – 3 | ESF – 5 |
| $G_{ESF}$ | $O_6$ | SF              | SF               | SF              | ESF – 2 | 0   |
| $G_{ESF-1}$ | $O_{h, 1}$ | SF              | SF              | SF              | ESF – 3 | 0   |
| $G_{ESF-2}$ | $O_{h, 2}$ | SF              | SF              | SF              | ESF – 3 | 0   |
| $G_{ESF-3}$ | $O_{h, 3}$ | SF              | SF              | SF              | ESF – 4 | 0   |
| $G_{ESF-4}$ | $O_{h, 4}$ | SF              | SF              | SF              | ESF – 4 | 0   |
| $G_{ESF-5}$ | $O_{h, 5}$ | SF              | SF              | SF              | ESF – 4 | 0   |
| $G_{SF}$ | $O_7$ | SF              | SF              | SF              | 0     |

https://doi.org/10.1371/journal.pone.0195204.1008
We firstly prove the indistinguishabilities between $G_{ESF}$, $G_{ESF-1}$, $G_{ESF-2}$ 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 $G_{F,0}, \cdots, G_{F,d,2}, \cdots, G_{F,d_2}$ where $G_{F,0} = G_{ESF}$ and $G_{F,d_2} = G_{ESF-1}$, 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 $i^{th}$ 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 $r, r', r'' \in Z_n$ randomly, and also chooses $X_1, X_2, Y_1, Y_2 \in G_{P_2}$ and $X_3, Y_3, Y_4 \in G_{P_3}$ randomly. It returns the group elements

$$\left\{(w'^{r}, g^{r'}, y^{r''} (u'h)^{r'} X_2 X_3, g'^{r} Y_2 Y_3, w'^{r}, g^{r'}, y^{r''} (u'^{h} h_0')^{r'} X_2 X_3, g'^{r} Y_2 Y_3)\right\}$$

(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 $G_{F,t,1}$ and $G_{F,t,2}$ with non-negligible advantage.

Let $Adv_A^{G_{F,t}}$ be the advantage of $A$ in a game $G_{F,t}$. From the Lemma 27, we obtain the following equation

$$Adv_A^{G_{F,t,1}} - Adv_A^{G_{F,t,2}} \leq \sum_{k=1}^{d_k} (Adv_A^{G_{F,t-k,1}} - Adv_A^{G_{F,t-k,2}}) \leq O(q_d)(Adv_A^{G_{F,t}})$$

(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 $g, u, h, v, w, u_0, h_0, v_0, w_0 \in G_{P_2}, g_2 \in G_{P_2}, g_3 \in G_{P_3}$. It chooses random exponents $s, \gamma, \delta_1, \delta_2, y, y_0, \sigma_1, \sigma_2, a', b', t', t'' \in Z_n$. It initially provides the attacker with the group elements:

$$(g, u, h, v, w, g^{(g_2 g_3)^\gamma}, w' (g_2 g_3)^{\gamma}, g' (g_2 g_3)^{\gamma}, v' (g_2 g_3)^{\gamma})$$

$$u_0, h_0, v_0, w_0, g_2^{(g_2 g_3)^{\gamma}}, g_2^{(g_2 g_3)^{\gamma}}, g_2^{(g_2 g_3)^{\gamma}}$$

(64)

What differs from the previous oracles here is the added $g_3^{\gamma}$ and term: notice that this is uniformly random in $G_{P_3}$, since $\gamma$ 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:

$$\left\{(w'^{r}, g^{r'}, y^{r''} (u'h)^{r'} X_2 X_3, g'^{r} Y_2 Y_3, w'^{r}, g^{r'}, y^{r''} (u'^{h} h_0')^{r'} X_2 X_3, g'^{r} Y_2 Y_3)\right\}$$

(65)

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

$$\left\{(w_0 g_2^{(g_2 g_3)^{\gamma}}, (g_2 g_3)^{\gamma}, (u_0 h_0')^{(g_2 g_3)^{\gamma}})\right\}$$

(66)

It is crucial to note that these $G_{P_3}$ terms are the same for each ciphertext-type query response.
Lemma 27 Under Assumptions 4, no PPT attacker can distinguish between $O_{\text{ESF-1}}$ and $O_{\text{ESF-2}}$ with non-negligible advantage. So no PPT attacker can distinguish between $G_{\text{ESF-1}}$ and $G_{\text{ESF-2}}$ with non-negligible advantage.

From the Lemma 27, we obtain the following equation

$$\text{Adv}_{\mathcal{A}}^{G_{\text{ESF-1}}} - \text{Adv}_{\mathcal{A}}^{G_{\text{ESF-2}}} \leq (\text{Adv}_{\mathcal{A}}^{\text{SA}})$$  \hspace{1cm} (67)

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

Oracle $O_{\text{ESF-3}}$ This oracle initializes in the same way with $O_{\text{ESF-2}}$ and provides the attacker the same initial elements as $O_{\text{ESF-2}}$. This oracle answers the ciphertext-type query and (H)IBE key-type query in the same way as $O_{\text{ESF-2}}$. To answer a challenge decryption key type query for $I$, it chooses random values $y, r, y', r' \in \mathbb{Z}_p, X_2, Y_2, X_3, Y_3 \in G_{p_1}$ randomly, and $X_0, Y_0, X_1, Y_1, X_2, Y_2 \in G_{p_3}$ and responds with:

$$w'(g_{g_{s,G_{1}}}y_0, g_{g_{s,G_{1}}}y, X_2, Y_2, Y_3, X_3)$$

and

$$w''(g_{g_{s,G_{1}}}y, g_{g_{s,G_{1}}}y', X_3, X_1, Y_1, Y_2)$$

Lemma 28 Under Assumptions 4, no PPT attacker can distinguish between $O_{\text{ESF-2}}$ and $O_{\text{ESF-3}}$ with non-negligible advantage. So no PPT attacker can distinguish between $G_{F-2,2}$ and $G_{F-2,2+1}$ with non-negligible advantage.

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

$$\text{Adv}_{\mathcal{A}}^{G_{\text{ESF-2}}} - \text{Adv}_{\mathcal{A}}^{G_{\text{ESF-3}}} \leq \sum_{h=1}^{q_k} (\text{Adv}_{\mathcal{A}}^{G_{F-2,h+1}} - \text{Adv}_{\mathcal{A}}^{G_{F-2,h}}) \leq O(q_k)(\text{Adv}_{\mathcal{A}}^{\text{SA}})$$  \hspace{1cm} (69)

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

Oracle $O'$ This oracle initializes in the same way as $O'$, and provides the attacker with initial group elements from the same distribution. It also responds to a ciphertext-type query as same as $O'$. It responds to the decryption-key-type and (H)IBE-key-type queries in the same way as $O_{\text{ESF-3}}$.

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

From the Lemma 29, we obtain the following equation

$$\text{Adv}_{\mathcal{A}}^{G_{\text{ESF-3}}} - \text{Adv}_{\mathcal{A}}^{G_{\text{ESF-4}}} \leq (\text{Adv}_{\mathcal{A}}^{\text{SA}})$$  \hspace{1cm} (70)

Indistinguishability of $G_{\text{ESF-4}}$ and $G_{\text{ESF-5}}$: We now prove the indistinguishability of $G_{\text{ESF-4}}$ and $G_{\text{ESF-5}}$ in a hybrid argument using polynomially many steps. We let $q_k$ denote the number of ciphertext-type queries made by a PPT attacker $\mathcal{A}$. Firstly we define hybrid games $L'_{-1,1}, L'_{0,2}, L'_{1,3}, L'_{0,1}, L'_{2,1}, L'_{3,2}, L'_{1,1}, \ldots, L'_{k, k-1}, L'_{k+1, k+1}, L'_{k+1, k}$, where $L'_{-1,1} = G_{\text{ESF-4}}$ and $L'_{k+1, k} = G_{\text{ESF-5}}$. 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 \( \bar{O}_q\) for each \( i \) from \( 0 \) to \( q_c - 1 \), and \( \bar{O}'_q\) for each \( i \) from \( 0 \) to \( q_c - 1 \).

Oracle \( \bar{O}_q \). This oracle acts the same as \( \bar{O}^*_q \) except that its response to the ciphertext-type query is as same as \( \bar{O}^*_q. \)

Oracle \( \bar{O}'_q \). This oracle acts the same as \( \bar{O}^*_q \) except that its response to the ciphertext-type query is as same as \( \bar{O}^*_q. \)

Lemma 30 Under Assumptions 3, no PPT attacker can distinguish between \( \bar{O}_q \) and \( \bar{O}^*_q \) with non-negligible advantage. So no PPT attacker can distinguish between \( L_{k,1} \) and \( L'_{k,1} \) with non-negligible advantage. We define additional oracles \( \bar{O}_q \) for each \( i \) from 0 to \( q_c - 1 \), and \( \bar{O}'_q \) for each \( i \) from 0 to \( q_c - 1 \).

Oracle \( \bar{O}_q \). This oracle acts the same as \( \bar{O}^*_q \) except that its response to the ciphertext-type query is as same as \( \bar{O}^*_q. \)

Oracle \( \bar{O}'_q \). This oracle acts the same as \( \bar{O}^*_q \) except that its response to the ciphertext-type query is as same as \( \bar{O}^*_q. \)

Lemma 31 Under Assumptions 4, no PPT attacker can distinguish between \( \bar{O}_q \) and \( \bar{O}^*_q \) with non-negligible advantage. So no PPT attacker can distinguish between \( L_{k,1} \) and \( L'_{k,1} \) with non-negligible advantage.

Lemma 32 Under Assumptions 3, no PPT attacker can distinguish between \( \bar{O}_q \) and \( \bar{O}^*_q \) with non-negligible advantage. So no PPT attacker can distinguish between \( L_{k,1} \) and \( L'_{k,1} \) with non-negligible advantage.

Let \( \text{Adv}^{L_{k,1}}_{\mathcal{A}}, \text{Adv}^{L'_{k,1}}_{\mathcal{A}}, \text{Adv}^{L'_{k,1}}_{\mathcal{A}} \) 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

\[
\text{Adv}^{G_{ESF-5}}_{\mathcal{A}} - \text{Adv}^{G_{ESF-5}}_{\mathcal{A}} \leq \sum_{k=0}^{q_c-1} (|\text{Adv}^{L_{k,1}}_{\mathcal{A}} - \text{Adv}^{L'_{k,1}}_{\mathcal{A}}| + |\text{Adv}^{L'_{k,2}}_{\mathcal{A}} - \text{Adv}^{L'_{k,3}}_{\mathcal{A}}| + |\text{Adv}^{L'_{k,3}}_{\mathcal{A}} - \text{Adv}^{L'_{k,1}}_{\mathcal{A}}|)
\]

\[
\leq q_c (2\text{Adv}^{G_{ESF-5}}_{\mathcal{A}} + \text{Adv}^{G_{ESF-5}}_{\mathcal{A}})
\]

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 \( J'_{1,1}, J'_{1,2}, \ldots, J'_{q_c,1}, J'_{q_c,2}, \ldots, J'_{q_c,2} \), where \( J'_{0,2} = G_{ESF-5} \) and \( J'_{q_c,2} = G_{SF}, \) 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 \( \bar{O}^*_0 \) and provides the attacker with initial group elements from the same distribution. Upon receiving a challenge decryption-key-type query for \( I, T \in \mathbb{Z}_n \), it chooses \( r, r', r'' \in \mathbb{Z}_n \) 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

\[
(w^{r'}(g g_3)^{r''}, g^{r''}(g g_3)^{r''}, e^{r''}(u'h)gX_3, g')X_3, g'Y_3,
\]

\[
w^{r'}(g g_3)^{r''}, g^{r''}(g g_3)^{r''}, e^{r''}(u'h)gX_3, g')X_3, g'Y_3
\]

(72)
to the attacker. It responds to a ciphertext-type query or a challenge (H)IBE-key-type query in the same way as $\mathcal{O}$. 

**Game** $J'_{h,1}$ This game $J'_{h,1}$ for $1 \leq h \leq q_d$ is almost the same as $G_{\text{ESP}-5}$ except the generation of decryption keys. The $i$th queried decryption key is generated as follows:
1. $i < h$: It generates a semi-functional $DK_{H(ID)^T}$.
2. $i = h$: It generates an ESF-5 $DK_{H(ID)^T}$.
3. $i > h$: It generates an ESF-4 $DK_{H(ID)^T}$.

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

**Lemma 33** Under Assumptions 4, no PPT attacker can distinguish between $\mathcal{O}$ and $O_{13/2}$ with non-negligible advantage. So no PPT attacker can distinguish between $J'_{h,1}$ and $J'_{h,2}$ 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 $\text{Adv}_{\mathcal{A}}^{G_{\text{ESP}},5}$ be the advantage of $\mathcal{A}$ in a game $G_{\text{ESP},5}$. From the Lemma 33, 34, we obtain the following equations

$$\text{Adv}_{\mathcal{A}}^{G_{\text{ESP}},5} - \text{Adv}_{\mathcal{A}}^{G_{\text{ESP}},4} \leq \sum_{h=1}^{q_d} (|\text{Adv}_{\mathcal{A}}^{J'_{h,1}} - \text{Adv}_{\mathcal{A}}^{J'_{h,2}}| + |\text{Adv}_{\mathcal{A}}^{J'_{h-2}} - \text{Adv}_{\mathcal{A}}^{J'_{h-1}}|)$$

$$\leq O(q_d)(\text{Adv}_{\mathcal{B}}^{G_{\text{SF}}} + \text{Adv}_{\mathcal{B}}^{G_{\text{SF}}})$$

So we obtain the following equation

$$\text{Adv}_{\mathcal{A}}^{G_{\text{ESP}}} - \text{Adv}_{\mathcal{A}}^{G_{\text{ESP},5}} \leq (|\text{Adv}_{\mathcal{A}}^{G_{\text{ESP}}} - \text{Adv}_{\mathcal{A}}^{G_{\text{ESP},4}}| +$$

$$\sum_{h=1}^{q_d} (|\text{Adv}_{\mathcal{A}}^{G_{\text{ESP},5}} - \text{Adv}_{\mathcal{A}}^{G_{\text{ESP},4}}| + |\text{Adv}_{\mathcal{A}}^{G_{\text{ESP},4}} - \text{Adv}_{\mathcal{A}}^{G_{\text{ESP},3}}|) +$$

$$\leq O(q_d)(\text{Adv}_{\mathcal{B}}^{G_{\text{SF}}} + (\text{Adv}_{\mathcal{B}}^{G_{\text{SF}}}) + O(q_d)(\text{Adv}_{\mathcal{B}}^{G_{\text{SF}}}) + (\text{Adv}_{\mathcal{B}}^{G_{\text{SF}}}) +$$

$$q_d(2\text{Adv}_{\mathcal{B}}^{G_{\text{SF}}} + \text{Adv}_{\mathcal{B}}^{G_{\text{SF}}}) + O(q_d)(\text{Adv}_{\mathcal{B}}^{G_{\text{SF}}} + \text{Adv}_{\mathcal{B}}^{G_{\text{SF}}})$$

$$\leq (O(q_d) + O(l))(\text{Adv}_{\mathcal{B}}^{G_{\text{SF}}} + \text{Adv}_{\mathcal{B}}^{G_{\text{SF}}})$$

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

$$\text{Adv}_{\mathcal{A}}^{G_{\text{ESP},5}} - \text{Adv}_{\mathcal{A}}^{G_{\text{ESP},4}} \leq (|\text{Adv}_{\mathcal{A}}^{G_{\text{ESP}} - p} - \text{Adv}_{\mathcal{A}}^{G_{\text{ESP}} - p,4}| +$$

$$|\text{Adv}_{\mathcal{A}}^{G_{\text{ESP}} - p,4} - \text{Adv}_{\mathcal{A}}^{G_{\text{ESP}} - p,3}| + |\text{Adv}_{\mathcal{A}}^{G_{\text{ESP}} - p,3} - \text{Adv}_{\mathcal{A}}^{G_{\text{ESP}} - p,2}|)$$

$$\leq (O(q_d) + O(l))(\text{Adv}_{\mathcal{B}}^{G_{\text{SF}}} + \text{Adv}_{\mathcal{B}}^{G_{\text{SF}}})$$

### 4.6 Indistinguishability of $G_C$ and $G_C'$ and Indistinguishability of $G_{\text{SF}}$ and $G_{\text{SF}}$

**Lemma 35** Under Assumptions 3 and 4, for any PPT attacker $\mathcal{A}$, the difference in $\mathcal{A}$’s advantage between $G_{\theta}$ 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 $B$ that breaks Assumption 3 with non-negligible advantage or a PPT algorithm $B$ that breaks Assumption 4 with non-negligible advantage.

While playing Game RHIBE$_0$ under Type-1 adversary, $A$ produces two values $I, r \in \mathbb{Z}_n$ which are unequal modulo $n$ but are equal modulo $p_3$, with non-negligible probability. We let $A$ denote $gcd(I - r, n)$, and we let $B$ denote $n/A$. We then have that $p_3$ divides $A$, and $B \neq 1$.

While playing Game RHIBE$_0$ under Type-2 adversary, $A$ produces two values $T, T' \in \mathbb{Z}_n$ 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 $B$ which breaks Assumption 3 with non-negligible advantage. $B$ receives $g, g_2, X_1, X_3, T$. It can use these terms to simulate Game RHIBE$_A$ with $A$ as follows. It picks values $a, b, c, d, a_0, b_0, c_0, d_0 \in \mathbb{Z}_N$ 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 $A$ the following public parameters:

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

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

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

\[ D_{0,0} = g^{\sum_{i=1}^{j} \gamma (X_i, g_2)}^{y_i \theta}, D_{0,1} = (X_i, g_2)_{\gamma}^{y_i \theta}, D_{0,2} = g^{\gamma \theta}, D_{0,3} = (X_i, g_2)_{\gamma}^{y_i \theta} (u_i^\gamma h_0)^{y_i \theta} \]

\[ D_{i,0} = g^{\gamma \theta} w_i^{\gamma \theta}, D_{i,1} = g^{\gamma \theta}, D_{i,2} = g^{\gamma \theta}, U_{i,3} = (u^\gamma h)^{\gamma \theta} w_i^{\gamma \theta}, i \in \{1, \ldots, j - 1\} \]

\[ D_{j,0} = g^{\gamma \theta} (X_i, g_2)^{y_j \theta}, D_{j,1} = (X_i, g_2)^{y_j \theta}, D_{j,2} = g^{\gamma \theta}, D_{j,3} = (X_i, g_2)^{y_j \theta} (u_j h)^{y_j \theta} \]

To make the semi-functional update key of $(I_1, \ldots, I_j)$ and $T$ for each $\theta$ with its value $\gamma_{\theta}$ in KUNode($BT_{ID_{\gamma_{\theta}}}, T, RL_{ID_{\gamma_{\theta}}}$), $B$ chooses random values $y_{i, \theta}, \ldots, y_{j-2, \theta}, y_{j-1, \theta}, r_0, \theta, r_1, \theta, \ldots, r_{j-1, \theta} \in \mathbb{Z}_n$. $B$ forms the challenge update key for $A$ as:

\[ U_{0,0} = g^{\gamma_{\theta} \gamma_{\theta}} \sum_{i=1}^{j-1} (X_i, g_2)^{y_i \theta}, U_{0,1} = (X_i, g_2)^{y_i \theta}, U_{0,2} = g^{\gamma_{\theta} \theta}, U_{0,3} = (X_i, g_2)^{y_i \theta} (u_i^\gamma h_0)^{y_i \theta} \]

\[ U_{i,0} = g^{\gamma_{\theta} \gamma_{\theta}} w_i^{\gamma_{\theta}}, U_{i,1} = g^{\gamma_{\theta} \theta}, U_{i,2} = g^{\gamma_{\theta} \theta}, U_{i,3} = (u^\gamma h)^{\gamma_{\theta} \theta} w_i^{\gamma_{\theta}}, i \in \{1, \ldots, j - 1\} \]

To make the semi-functional secret key of $(I_1, \ldots, I_j)$ and $T$ for each $\theta$ with its value $\gamma_{\theta}$ in Path($ID_{\theta}$), $B$ chooses random values $y_{i, \theta}, \ldots, y_{j-1, \theta}, y_j', r_0, \theta, \ldots, r_{j}, \theta \in \mathbb{Z}_n$. $B$ forms the
challenge secret key for \( A \) as:

\[
K_{i,0} = g^{\sigma_0} \cdot w^{\sigma}, \quad K_{i,1} = g^{\sigma_0}, \quad K_{i,2} = g^{\sigma_0}, \quad K_{i,3} = (u^\theta h)^{-w^{\sigma}}, \quad i \in \{1, \ldots, j - 1\}
\]

\[
K_{i,0} = g^{w^{\sigma}} \cdot \sum_{i=1}^{j-1} (X_i X_3 g_i)^{\theta_i} \cdot K_{i,3} = (X_i X_3 g_i)^{\theta_i}, \quad K_{i,2} = g^{\sigma},
\]

\[
K_{i,3} = (X_i X_3 g_i)^{\theta_i} (u^\theta h)^{\sigma_i}
\]

So \( B \) can play Game RHIBE\(_{SF}\) and Game RHIBE\(_{SF}\) with \( A \). Now, if \( A \) 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 \( B \) guesses randomly. However, with non-negligible probability, \( A \) will produce such an \( I, I' \) or \( T, T' \). \( B \) 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 \( (X_i X_3)^B \neq 1 \) (this confirms that \( p_3 \) does not divide \( B \), hence it must divide \( A \)). When \( B \) detects this situation, it can test whether \( T \in G_{p_1} \) or \( T \in G_{p_0, p_3} \) by testing if \( T^B = 1 \) holds. If \( T^B = 1 \) holds, then \( T \in G_{p_0} \). If \( T^B \neq 1 \), then \( T \in G_{p_0, p_3} \). Thus, \( B \) achieves non-negligible advantage in breaking Assumption 3.

If case 2) occurs with non-negligible probability, we can create a \( B \) which breaks Assumption 4 with non-negligible advantage. \( B \) receives \( g, g_0, X_1 X_2, Y_2 Y_3, T \). It can use these terms to simulate Game RHIBE\(_{SF}\) with \( A \) as follows. It gives \( A \) the public parameters like in the case 1. We note that \( B \) knows the master secret key \( \sigma \), so it can easily make normal keys.

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

\[
C = Me(X_i X_2, g)^{t_i}, \quad C_0 = X_i X_2,
\]

\[
C_{0,1} = (X_i X_2)^{t_0}, \quad C_{0,2} = g^{t_0 + h_0}, \quad C_{0,3} = g^h
\]

\[
C_{i,1} = (X_i X_2)^{t_i} g^\sigma, \quad C_{i,2} = g^{t_i + h_i}, \quad C_{i,3} = g^i, \quad i \in \{1, \cdots, j\}
\]

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 \( \{I_1, \cdots, I_j\} \), \( B \) 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:

\[
D_{0,0} = g^{-\sum_{i=1}^{j} w^{\sigma} (Y_2 Y_3)^{h_i w^{\sigma}}}, \quad D_{0,1} = g^{0} (Y_2 Y_3)^{h_0}, \quad D_{0,2} = g^{0}, \quad D_{0,3} = g^{0} (Y_2 Y_3)^{h_0 w^{\sigma} (u^\theta h)^h}
\]

\[
D_{i,0} = g^{w^{\sigma}}, \quad D_{i,1} = g^{0}, \quad D_{i,2} = g^{i}, \quad D_{i,3} = (u^\theta h)^{-w^{\sigma}}, \quad i \in \{1, \cdots, j - 1\}
\]

\[
D_{j,0} = g^{w^{\sigma}}, \quad D_{j,1} = g^{0}, \quad D_{j,2} = g^{i}, \quad D_{j,3} = g^i, \quad D_{j,3} = g^{0} (Y_2 Y_3)^{h_i w^{\sigma} (u^\theta h)^h}
\]

To make the semi-functional update key of \( \{I_1, \cdots, I_{j-1}\} \) and \( T \) for each \( \theta \) with its value \( \gamma_\theta \) in KUNode(\( BT_{ID_{j-1}} \), \( T \), RL\(_{ID_{j-1}}\)), \( B \) chooses random values \( y_{0,\theta}, y_{1,\theta}, \cdots, y_{j-1,\theta} \)
and G

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

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

(75)

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

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

\[ CT = (M_\beta e(g, T)^{\beta_1}, T, T^{t_0}v_0^{t_0}, (u_0^{t_0}h_0)^{t_0}, g^{c_1}, \{T^{t_l}v_l, (u_0^{t_l}h_0)^l, g^{c_l}\}) ]  

(76)

This implicitly sets g^e equal to the G_{p_1} part of T. If T \in G_{p_1}, then this is a well-distributed normal ciphertext, and B has properly simulated Game G_{Real}. If T \in G_{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, $B$ has properly simulated Game $G_{C}$ in this case. Thus, $B$ can use the output of $A$ 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 $A$ who achieves a non-negligible advantage in Game RHIBE$_{SF}$. We will create a PPT algorithm $B$ which has a non-negligible advantage against Assumption 2.

$B$ receives $g, g_{2}, g_{3}, g^{f} X_{2}, g^{f} Y_{2}, T$. It chooses $a, b, c, d, a_0, b_0, c_0, d_0$ randomly from $Z_p$ and sets $u = g^a, v = g^b, w = g^c, w_0 = g^{a_0}, v_0 = g^{b_0}, w_0 = g^{d_0}$. It gives the public parameters

$$PP = (g, u, h, v, w, u_0, h_0, v_0, w_0, \Omega = e(g, g^{f} X_{2}))$$

(77)

to $A$. We note that $B$ does not know the master secret key $\alpha$. For a secret key query for $(I_0, \cdots, I_k)$, $B$ will create a semi-functional secret key as follows. It chooses $f_1$ randomly and $r_{1, \theta}, \cdots, r_{k, \theta}, b_{1, \theta}, \cdots, b_{k, \theta} \in Z_p$ randomly for each node $\theta \in Path(ID_0)$. The semi-functional secret key $SK_{ID_0}$ is formed as $((\theta, PSK_{\theta}|_{\theta \in path})$, in which

$$PSK_{\theta} = \left\{ \left( K_{1,0}, K_{1,1}, K_{1,2}, K_{1,3} \right) \right\}$$

and we have $(K_{1,0}, K_{1,1}, K_{1,2}, K_{1,3})$ as

$$\begin{align}
&\left( \begin{array}{c}
\left( g^{e^{i}} w_{0}^{\theta}, g^{h_{0}^{\theta}}, g^{f_{0}^{\theta}}, (u^{b_{0}} h)^{\psi_{0}^{\theta}} v^{\psi_{0}^{\theta}},
\end{array} \right),
&0 \leq i \leq k - 1
\end{align}$$

$$\left( g^{e^{i}} \prod_{i=1}^{k-1} h_{i} w_{i}^{\theta} (g_{2,3})^{f_{i}(d_i+1)}, (g_{0,3})^{f_{i}(d_i+1)}, g_{1,3}^{f_{i}}, (u h)^{\psi_{i}} v^{\psi_{i}} (g \theta_{i}^{f_{i}})^{\psi_{i}}
\right),
\quad i = k$$

(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 $T$, $B$ generates a semi-functional update key as follows. It chooses $r_{j, \theta}, \cdots, r_{j, \theta}, b_{j-1, \theta}, b_{j, \theta} \in Z_p$ randomly for each node $\theta \in KUNode (BT_{ID_0})$ and $f_2$ randomly. And it will implicitly set $b_{0, \theta} = (b_{0, \theta} + \chi) mod p_1$. The semi-functional update key is formed as $UK_{ID_0,T} = ((\theta, TUK_{\theta}|_{\theta \in KUNode})$ and

$$TUK_{\theta} = \left\{ \left( U_{b_{0}, U_{1,1}, U_{1,2}, U_{1,3}} \right) \right\}_{j=0}^{j-1}$$

$$U_{0,0} = g^{e^{0}} \prod_{i=1}^{j-1} h_{i} w_{0}^{\theta} (g_{2,3})^{f_{i}(d_i+1)}, U_{0,1} = g^{f_{1}} X_{2,0} g^{h_{0}^{\theta}} (g_{2,3})^{f_{1}},$$

$$U_{i,0} = g^{e^{i}} w_{0}^{\theta}, U_{i,1} = g^{h_{0}^{\theta}}, U_{i,2} = g^{f_{0}^{\theta}}, U_{i,3} = (u h)^{\psi_{i}} v^{\psi_{i}}, i \in \{1, \cdots, j - 1\}$$

(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 $y_2 = f_2 (mod p_2), y_2 = (f_2 + log \theta g_{2,3}) (mod p_2)$, then $X_{2,0} (g_{2,3})^{f_{2}} = X_{2,0} (g_{2,3})^{f_{2}}, (g_{2,3})^{f_{2}} = (g_{2,3})^{f_{2}}$. 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 $T$, $B$ 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
\[
D_{0,0} = g^{\Pi^{l-1} (g^sX_2)^{d_{i+1}} w_0^\nu (g_{6,8})^{\nu_{d_{i+1}}}}, D_{0,1} = (g^sX_2)g^{h^\nu (g_{6,8})^{\nu_2}},
\]
\[
D_{0,2} = g^\nu, D_{0,3} = (g^sX_2)^{\nu_0} (u_0^\nu h_0)^{\nu_0} v_0^\nu (g_{6,8})^{\nu_0},
\]
\[
D_{i,0} = g^{\nu_1} w_i^\nu, D_{i,1} = g^{\nu_1}, D_{i,2} = g^{\nu_1}, D_{i,3} = (u^\nu h^\nu)^{\nu_i}, i \in \{1, \ldots, j - 1\}
\]
\[
D_{j,0} = g^{\nu_1} w_i^\nu (g_{6,8})^{\nu_i}, D_{j,1} = g^{\nu_1} (g_{6,8})^{\nu_i},
\]
\[
D_{j,2} = g^{\nu_1}, D_{j,3} = (u^\nu h^\nu)^{\nu_i} (g_{6,8})^{\nu_i}
\]
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 \( (I_1, \ldots, I_t) \) and a challenge time \( T \). \( \mathcal{B} \) creates the challenge ciphertext as follows. It chooses \( t_1, \ldots, t_i, \delta_1, \delta_2 \) randomly from \( \mathbb{Z}_n \) and \( \beta \) randomly from \( \{0, 1\} \) and sets:
\[
C = M_\beta T, C_0 = g^sY_2,
\]
\[
C_{0,1} = (g^sY_2)^{\delta_1} g^s_i, C_{0,2} = (u_i^\nu h_0)^{\nu_0}, C_{0,3} = g^{\nu_0}
\]
\[
C_{1,2} = (g^sY_2)^{\delta_1} g^s_i, C_{1,2} = (u_i^\nu h_0)^{\nu_0}, C_{1,3} = g^{\nu_0}, i \in \{1, \ldots, l\}
\]
If \( T = e(g, g)^{\nu_{\gamma}} \), this is a well-distributed semi-functional encryption of \( M_\beta \) with \( \gamma = \log_{a_2} Y_2, \delta_1 = d \cdot \log_{a_2} Y_2 + \delta_1, \delta_2 = d_0 \cdot \log_{a_2} Y_2 + \delta_2 \). Notice that \( \delta_1 \) and \( \delta_2 \) 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 \( \beta \). 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 re-design 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 Definition of the ephemeral semi-functional ciphertexts and keys

In the definition 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 \( G_{p, p_2} \), and only the first elements of the rest of ciphertext-element-groups are added by \( G_{p_1} \) terms. In the definition of the third type of ephemeral semi-functional
ciphertext, every element of the first $i-1$ ciphertext-element-groups is in $G_{p,p^{2p_0}}$ for the $i^{th}$ ciphertext-element-group, the first element is in $G_{p,p^{2p_3}}$, its rest elements are in $G_{p,p^{3}}$, and for the rest ciphertext-element-groups, we add $G_{p}$ terms on the first elements of them. In the definition of the fourth type of ephemeral semi-functional ciphertext, every elements of the first $i-1$ ciphertext-element-groups are in $G_{p,p^{2p_2}}$, every elements of the $i^{th}$ ciphertext-element-group are in $G_{p,p^{2p_3}}$, and for the rest ciphertext-element-groups, we add $G_{p}$ terms on the first elements of them. In the definition of the fifth type of ephemeral semi-functional ciphertext, every element of all ciphertext-element-groups is in $G_{p,p^{2p_3}}$.

EncryptESF-1 ($\text{ID}, T, M, PP, \sigma, \delta$) $\rightarrow \widetilde{CT}_{E-1}$. Let the normal ciphertext be
\[
CT_{\text{ID}} = (C, C_0, \{C_{i,1}, C_{i,2}, C_{i,3}\}_{i=0}^l).
\]
It chooses $\gamma, \delta, a, b$, and random $t_0, \ldots, t_l \in \mathbb{Z}_n$ and forms the ESF-1-CT $\widetilde{CT}_{E-1}$ as
\[
\left\{\left(C, C_0, g, g^{\gamma+t_0}, g^{\gamma+2t_0}, \ldots, g^{\gamma+lt_0} \right), C_{i,0,1}, C_{i,0,2}, C_{i,0,3}, \{C_{i,1}, C_{i,2}, C_{i,3}\}_{i=0}^l\right\}
\]
EncryptESF-2 ($\text{ID}, T, M, PP, \sigma, \delta,\gamma, \delta, a, b$) $\rightarrow \widetilde{CT}_{E-2}$. It chooses $\gamma, \delta, a, b$, and random $t_0, \ldots, t_l \in \mathbb{Z}_n$. It forms the first two elements and the first $k$ element-groups of ESF-2-CT as same as of ESF-1-CT, and the rest element-groups of ESF-2-CT as same as of SF-CT.

EncryptESF-3 ($\text{ID}, T, M, PP, \sigma, \delta,\gamma, \delta, a, b$) $\rightarrow \widetilde{CT}_{E-3}$. It chooses $\gamma, \delta, a, b$, and random $t_0, \ldots, t_l \in \mathbb{Z}_n$. It forms the first two elements and the first $k-1$ element-groups of ESF-3-CT as same as of ESF-1-CT, and the $k^{th}$ element-group of ESF-3-CT as
\[
\left\{\left(C, C_0, g, g^{\gamma+t_0}, \ldots, g^{\gamma+t_l} \right), C_{i,0,1}, C_{i,0,2}, C_{i,0,3}, \{C_{i,1}, C_{i,2}, C_{i,3}\}_{i=0}^{k-1}\right\}
\]
and the rest element-groups of ESF-3-CT as same as of SF-CT.

EncryptESF-4 ($\text{ID}, T, M, PP, \sigma, \delta,\gamma, \delta, a, b$) $\rightarrow \widetilde{CT}_{E-4}$. It chooses $\gamma, \delta, a, b$, and random $t_0, \ldots, t_l \in \mathbb{Z}_n$. It forms the first two elements and the first $k-1$ element-groups of ESF-4-CT as same as of ESF-1-CT, and the $k^{th}$ element-group of ESF-4-CT as
\[
\left\{\left(C, C_0, g, g^{\gamma+t_0}, \ldots, g^{\gamma+t_l} \right), C_{i,0,1}, C_{i,0,2}, C_{i,0,3}, \{C_{i,1}, C_{i,2}, C_{i,3}\}_{i=0}^{k-1}\right\}
\]
and the rest element-groups of ESF-4-CT as same as of SF-CT.

EncryptESF-5 ($\text{ID}, T, M, PP, \sigma, \delta$) $\rightarrow \widetilde{CT}_{E-5}$. Let the normal ciphertext be
\[
CT_{\text{ID}} = (C, C_0, \{C_{i,1}, C_{i,2}, C_{i,3}\}_{i=0}^l).
\]
It chooses $\gamma, \delta, a, b$, and random $t_0, \ldots, t_l, t'_0, \ldots, t'_1, t''_0, \ldots, t''_1 \in \mathbb{Z}_n$. It forms the first two elements of $\widetilde{CT}_{E-5}$ as $C, C_0, g$, and forms the element-groups of ESF-5-CT as
\[
\left\{\left(C, C_0, g, g^{\gamma+t_0}, \ldots, g^{\gamma+t_l} \right), C_{i,0,1}, C_{i,0,2}, C_{i,0,3}, \{C_{i,1}, C_{i,2}, C_{i,3}\}_{i=0}^{k-1}\right\}
\]
In the definition of the first type of ephemeral semi-functional secret key, we add $G_{p}$ term on the last 2 elements of the last element-group. In the definition of the second type of ephemeral semi-functional secret key, we add $G_{p,p^{3}}$ term on the last 2 elements of the last element-group. In the definition of the third type of ephemeral semi-functional secret key, we add $G_{p}$ term on the first 2 elements of the last element-group and add $G_{p,p^{3}}$ term on the last 2 elements of the last element-group. In the definition of the fourth type of ephemeral semi-functional secret key, every element of the last element-group is in $G_{p,p^{2p_3}}$. In the definition 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 $G_{p_2}$, and the third element of the last element-group is in $G_{p_3}$.

**SKKeyESF-1** $(ID, ST_{ID_{j-1}}, PP, \theta) \rightarrow \widetilde{PSK}_{E-1}$ Let the correlative component key to the node $\theta \in Path(ID)$ in the BTide be $PSK_{ID_{j-1}} = (\{K_{i_0}, K_{i_1}, K_{i_2}, K_{i_3}\})$. It chooses random values $X_4, Y_3 \in G_{p_3}$ and forms the component ESF-1-ESK by changing the last element-group as

$$(K_{j_0}, K_{j_1}, K_{j_2}, Y_3, K_{j_3}, X_3)$$

**SKKeyESF-2** $(ID, ST_{ID_{j-1}}, PP, \theta) \rightarrow \widetilde{PSK}_{E-2}$ Let the correlative component key to the node $\theta \in Path(ID)$ in the BTide be $PSK_{ID_{j-1}} = (\{K_{i_0}, K_{i_1}, K_{i_2}, K_{i_3}\})$. 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-PSK by changing the last element-group as

$$(K_{j_0}, K_{j_1}, K_{j_2}, Y_2, Y_3, K_{j_3}, X_3)$$

**SKKeyESF-3** $(ID, ST_{ID_{j-1}}, PP, \theta) \rightarrow \widetilde{PSK}_{E-3}$ 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 secret key ESF-3-PSK by constructing $\kappa(I_{p}, y', r)$ in the last element-group as

$$(w', g_{y'^2}, g_{y'}, g_1^2, g_1^{'y'}, g'^y_2 Y_3, Y_2, Y_3, r)(u'^{y')X_2 X_3)}$$

And the contraction of the other element-groups follows the construction of $SK_{HIBES_0}$ in RHIBE.GenKey.

**SKKeyESF-4** $(ID, ST_{ID_{j-1}}, PP, \theta) \rightarrow \widetilde{PSK}_{E-4}$ 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-PSK by constructing $\kappa(I_{p}, y', r)$ in the last element-group as

$$(w', (g_{g_2} y'^2), g_{g_2} y', g_2^y g_1^2, g_2^{'y'}, g'^y_2 Y_3, Y_2, Y_3, r)(u'^{y')X_2 X_3)}$$

And the contraction of the other element-groups follows the construction of $SK_{HIBES_0}$ in RHIBE.GenKey.

**SKKeyESF-5** $(ID, ST_{ID_{j-1}}, PP, \theta) \rightarrow \widetilde{PSK}_{E-5}$ 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-PSK by constructing $\kappa(I_{p}, y', r)$ in the last element-group as

$$(w', (g_{g_5} y'^2), g_{g_5} y', g_5^y g_1^2, g_5^{'y'}, g'^y_2 Y_3, Y_2, Y_3, r)(u'^{y')X_2 X_3)}$$

And the contraction of the other element-groups follows the construction of $SK_{HIBES_0}$ 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.

**UKKeyESF-1** $(T, ST_{ID_{j-1}}, RL_{ID_{j-1}}, PP, \theta) \rightarrow \widetilde{TUK}_{E-1}$ Let the correlative component key to the node $\theta \in KU\text{Node}(RL_{ID_{j-1}}, T)$ be $TUK_{ID_{j-1}} = (\{U_{i_0}, U_{i_1}, U_{i_2}, U_{i_3}\}^{-1})$. It chooses random values $r_1^*, r_2^* \in Z_n$ and forms the ephemeral semi-functional secret key $\widetilde{TUK}_{E-1}$ by changing the first element group as it chooses random values $X_5, Y_5 \in G_{p_5}$ and forms the component
ESF-1-SK $\widetilde{\text{TU}K}_{E-1}$ by changing the first element-group as

\[(U_{0}, U_{1}, U_{2}Y_{3}, U_{2}X_{3})\]

**UKeyESF-2** ($T, ST_{ID_{1}}, RL_{ID_{1},T}, PP, \theta) \rightarrow \widetilde{\text{TU}K}_{E-2}$ Let the correlative component key to the node $\theta \in K\text{UNode}(RL_{ID_{1},T})$ be $\text{TU}K_{ID_{1},T,\theta} = (\{U_{0}, U_{1}, U_{2}, U_{3}\})$. It chooses random values $\tilde{r}_{1}, \tilde{r}_{2} \in Z_{\pi}$ and forms the ephemeral semi-functional secret key $\widetilde{\text{TU}K}_{E-2}$ by changing the first element group as

\[(U_{0}, U_{0,1}(g_{2}g_{1})^{\tilde{r}_{1}}, U_{2,0}(g_{2}g_{1})^{\tilde{r}_{2}})\]

**UKeyESF-3** ($T, ST_{ID_{1}}, RL_{ID_{1},T}, PP, \theta) \rightarrow \widetilde{\text{TU}K}_{E-3}$ It chooses the values $y', r \in Z_{p}$ randomly, $X_{2}, Y_{2} \in G_{P_{2}}$ randomly, and $X_{3}, Y_{3} \in G_{P_{3}}$ randomly and forms the component secret key ESF-3-UK $\widetilde{\text{TU}K}_{E-3}$ by constructing $\kappa_{T}(y', r)$ of the first element-group as

\[(w_{0}^{\tilde{r}_{1}}g_{2}^{\tilde{r}_{1}}, g_{1}^{\tilde{r}_{1}}, g_{1}^{\tilde{r}_{1}}Y_{2}Y_{3}, v_{0}^{\tilde{r}_{1}}(u_{0,1}h_{0})X_{2}X_{3})\]

And the construction of the other element-groups follows the construction of RSK_{HIBE} and SK_{HIBE,S} in RIBE.UpdateKey.

**UKeyESF-4** ($T, ST_{ID_{1}}, RL_{ID_{1},T}, PP, \theta) \rightarrow \widetilde{\text{TU}K}_{E-4}$ It chooses the values $y', r \in Z_{p}$ 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 $\widetilde{\text{TU}K}_{E-4}$ by changing $\kappa_{T}(y', r)$ in the first element-group as

\[(w_{0}^{\tilde{r}_{1}}(g_{2}g_{1})^{\tilde{r}_{1}}, g_{1}^{\tilde{r}_{1}}(g_{2}g_{1})^{\tilde{r}_{1}}, g_{1}^{\tilde{r}_{1}}Y_{2}Y_{3}, v_{0}^{\tilde{r}_{1}}(u_{0,1}h_{0})X_{2}X_{3})\]

And the construction of the other element-groups follows the construction of RSK_{HIBE} and SK_{HIBE,S} in RIBE.UpdateKey.

**UKeyESF-5** ($T, ST_{ID_{1}}, RL_{ID_{1},T}, PP, \theta) \rightarrow \widetilde{\text{TU}K}_{E-5}$ It chooses the values $y', r \in Z_{p}$ 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 $\widetilde{\text{TU}K}_{E-5}$ by changing $\kappa_{T}(y', r)$ in the first element-group as

\[(w_{0}^{\tilde{r}_{1}}(g_{2}g_{1})^{\tilde{r}_{1}}, g_{1}^{\tilde{r}_{1}}(g_{2}g_{1})^{\tilde{r}_{1}}, g_{1}^{\tilde{r}_{1}}Y_{3}, v_{0}^{\tilde{r}_{1}}(u_{0,1}h_{0})X_{2}X_{3})\]

And the construction of the other element-groups follows the construction of RSK_{HIBE} and SK_{HIBE,S} in RIBE.UpdateKey.

**DKeyESF-i** ($ID_{i}, T, MSK, RL_{ID_{i},T}, PP) \rightarrow \widetilde{\text{DK}_{E}}$ The ephemeral semi-functional decryption key generation algorithm firstly retrieves $\theta \in K\text{UNode}(RL_{ID_{i},T})$ and gets $\text{TU}K_{E}$ and $PSK_{E}$ which are the correlative subkey to the node $\theta^{'},$ from UKeyESF-i ($T, ST_{ID_{i},T}, \theta^{'})$ and SKKeyESF-i ($ID_{i}, \theta^{'})$, and then forms the ephemeral semi-functional decryption key $\widetilde{\text{DK}_{E}}$ 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 $H_{h_{1},2}$ and $H_{h_{1},1}$ with non-negligible advantage.

**Proof** We assume $B$ interacts with one of $O_{0}, O_{1/2}$. $O$ receives $g, g_{2}, X_{1}X_{3}, Y_{1}Y_{3}, T$. $O$ will simulate either $O_{0}$ or $O_{1/2}$ with $B$, depending on the value of $T$ (which is either in $G_{P_{2}}$ or $G_{P_{1},P_{3}}$). $O$ picks values $a, b, c, d, a_{0}, b_{0}, c_{0}, d_{0} \in Z_{N}$ uniformly at random and sets $u = g^{a}$, $h = g^{b}$, $v = g^{c}$, $v^{2} = g^{d}$.
some time

2. \( \log \) of

by the Chinese Remainder Theorem, and

3. \( i \)

Eq 82 that have been offered by the oracle simulator.

1. \( T \)

as same as Eq 7 to

\( g \), \( B \)

chooses

We note that these are properly distributed, with

where (\( y \), \( g \), \( B \), \( A \))

requests the challenge ciphertext for messages

\( g \), \( v \).

Then \( T \) makes a ciphertext-type query for some identity \( \Gamma \), \( O \) responds by choosing a random \( t \) in \( Z_N \) and returning

\( (w^g_{X_1'}, v', g', (v^w h)) \) to \( B \) as same as Eq 6. When \( B \) makes a ciphertext-type query for some time \( T \), \( O \) responds by choosing a random \( t_0 \) in \( Z_N \) and returning

\( (w^g_{X_1'}, v_0', g'^v, (u^g_{h_0} h^v)) \)

as same as Eq 7 to \( B \). Then \( B \) creates the semi-functional ciphertexts as

\[
CT_{\alpha \beta} = (g^w_{X_1'}, v', g', (v^w h)^v, \{w^g_{X_1'}, v_0', (u^g_{h_0} h)^v, g'^v\}_t).
\]

When \( B \) creates the HIBE private key with the index pair \( (h, i) \) for some identity vector \( (I_1, \cdots, I_f) \) in the index \( h \) node, the HIBE private key with an index pair \( (h, i) \) is generated as follows:

1. \( i_0 < h \): It randomly chooses \( y_1, \cdots, y_{j-1}, *_1, \cdots, *_{j-1}, r_1, \cdots, r_{j-1}, r'_1, z, z' \in Z_n \) and generates an ESF-2 SK \( SK_{IBE, h, 0} \),

\[
K_{i_0} = g^{y_0 w_0}, K_{i_1} = g^{y_1}, K_{i_2} = g^{y_2}, K_{i_3} = (u^w h)^w, i \in \{1, \cdots, j-1\}
\]

\[
K_{j_0} = g^{w^{j_0 w_0}} \sum_{i=1}^{j-1} w_i K_{i_0} = g^{y_0}, K_{j_2} = v^w (X_{j_1} X_{j_2})^{r_{(a+j)+b}} g^v, K_{j_3} = (X_{j_1} X_{j_2})^{r_{j} g'^v}
\]

It implicitly sets \( g' \) to be \( X_{j_1}' \) and that is a properly distribution ESF-2 SK.

2. \( i_0 = h \): \( B \) chooses random values \( y_1, \cdots, y_{j-1}, *_1, \cdots, *_{j-1}, r_1, \cdots, r_{j-1} \in Z_n \). \( B \) forms the challenge key as:

\[
K_{i_0} = g^{y_0 w_0}, K_{i_1} = g^{y_1}, K_{i_2} = g^{y_2}, K_{i_3} = (u^w h)^w, i \in \{1, \cdots, j-1\}
\]

\[
K_{j_0} = g^{w^{y_0 w_0}} \sum_{i=1}^{j-1} T_i, K_{j_1} = T_1, K_{j_2} = T_2, K_{j_3} = T_3
\]

where \( (T_0, T_1, T_2, T_3) \) is the challenge HIBE key queried to \( O \) who chooses a random \( y_0 \in Z_n \) and returns

\( (w', g', v', (v^w h)^v, (u^g_{h_0} h^v)) \) to \( B \).

3. \( i_0 > h \): It simply generates a normal HIBE private key.
In the challenge HIBE key, it implicitly sets \( g^v \) to be the \( G_p \) part of \( T \). If \( T \in G_p \), then this matches the distribution of \( O_0 \) (since there are no \( G_p \) terms here), and so this will be a properly distributed normal key and \( B \) is playing Game \( H_{b_{-1,2}} \). If \( T \in G_0 \), 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 \( G_p \) terms attached to the last two group elements) and then \( B \) is playing Game \( H_{b_{-1,1}} \).

Hence, if a PPT attacker can distinguish between \( H_{b_{-1,2}} \) and \( H_{b_{-1,1}} \) with non-negligible advantage, \( O \) can distinguish between \( O_0 \) and \( O_{1/2} \) with non-negligible advantage. It means \( O \) 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 \( H_{b_{-1,2}} \) and \( H_{b_{-1,1}} \) 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 \( H_{b_{-1,1}} \) and \( H_{b_{-2,1}} \) with non-negligible advantage.

**Proof** We assume \( B \) interacts with one of \( O_{1/2} \) and \( O_1 \). \( O \) receives \( g, g_b, X_1 X_2 Y_2 Y_3, T \). \( O \) will simulate either \( O_{1/2} \) or \( O_1 \) with \( B \), depending on the value of \( T \) (which is either in \( G \) or \( G_0 \)). \( O \) picks values \( a, b, c, d, a_0, b_0, c_0, d_0 \in Z_N \) uniformly at random and sets \( u = g^a, h = g^b, v = g^c, w = g^d, u_0 = g^{a_0}, h_0 = g^{b_0}, v_0 = g^{c_0}, w_0 = g^{d_0}. \) \( B \) initially obtains the group elements

\[
\begin{align*}
g, u, h, v, w, X_1 X_2, w^\alpha (Y_2 Y_3)^{v\alpha}, g^\beta (Y_2 Y_3)^{v\beta}, \psi (Y_2 Y_3)^{v\psi}, & \\
u_0, h_0, v_0, w_0, (Y_2 Y_3)^{v\alpha}, g^{h_0} (Y_2 Y_3)^{v\beta}, v_0^\alpha (Y_2 Y_3)^{v\beta} &
\end{align*}
\]

from its oracle simulator, where \( y, \psi_1, \psi_2, \alpha, \beta \in Z_n \) are randomly chosen. It chooses \( \alpha \in Z_n \) randomly, and gives \( A \) the public parameters in Eq 83. When \( A \) requests the challenge ciphertext for messages \( M_{0i}, M_1 \), identity vector \( (I_1^1, \cdots, I_1^n) \) and \( T^t, B \) makes a ciphertext-type query to the oracle for each \( I_i^t \) and \( T^t \). In response to each query for \( I_i^t \), \( O \) responds \((X_1 X_2)^{v_i}, g^\alpha (u^\beta h)^\beta) \) to \( B \) by choosing a random \( t_i \in Z_N \). In response to the query for \( T^t \), \( O \) responds \((X_1 X_2)^{v_i}, g^\alpha (u_0^\beta h_0)^\beta) \) to \( B \) by choosing a random \( t_0 \in Z_N \). Then \( B \) creates the semi-functional ciphertexts successfully.

When \( B \) creates the HIBE private key with the index pair \((h, i_0)\) for some identity vector \((I_1, \cdots, I_1)\) in the index node, the HIBE private key with an index pair \((h, i_1)\) is generated as follows:

1. \( i_c < h_c \); \( B \) chooses random values \( y_1, \cdots, y_p, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_p, z, z' \in Z_n \) and generates a ESF-2 PSK_{HIBE,hr} 

\[
\begin{align*}
K_{00} &= g^{h_0} w_0, K_{01} = g^{h_1}, K_{11} = g^{h_2} = (u^\beta h)^\beta, K_{13} = (u^\beta h)^\beta w, i \in \{1, \cdots, j - 1\} \\
K_{j0} &= g^{y_j} \sum_{i=0}^{j-1} (w_i, K_{j1} = g^{h_i}, K_{j2} = v^\alpha (u^\beta h)^\beta (Y_2 Y_3)^{v\beta}, K_{j3} = g^{h_i} (Y_2 Y_3)^{v\beta}
\end{align*}
\]

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

\[
\begin{align*}
K_{00} &= g^{h_0} w_0, K_{01} = g^{h_1}, K_{11} = g^{h_2} = (u^\beta h)^\beta, K_{13} = (u^\beta h)^\beta w, i \in \{1, \cdots, j - 1\} \\
K_{j0} &= g^{y_j} \sum_{i=0}^{j-1} T_i, K_{j1} = T_j, K_{j2} = T_j, K_{j3} = T_2
\end{align*}
\]
from its oracle simulator. It chooses $i$ terms will have random elements in both $G_p$ attached (matching the distribution of $O_{1/2}$) and then $B$ is playing Game $H_{b_{1,1}}$. And when $T \in G$, these last two terms will have random elements in both $G_p$ and $G_p$ attached (matching the distribution of $O_1$) and then $B$ is playing Game $H_{b_{1,2}}$.

Hence, if a PPT attacker can distinguish between $H_{b_{1,1}}$ and $H_{b_{1,2}}$ 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 $H_{b_{1,1}}$ and $H_{b_{1,2}}$ 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 $B$ interacts with one of $O_{k-1}, O_k$. $\mathcal{O}$ receives $g, g_2, X, X_y, Y, Y_y, T$. $\mathcal{O}$ will simulate either $O_{k-1}$ or $O_k$ with $B$, depending on the value of $T$ (which is either in $G_{p_1}$ or $G_{p_1, p_3}$). $B$ initially obtains the group elements in Eq 82

$$g, u, h, v, w, g^{ug_2}, (X, X_y)^{vdj_0}, (X, X_y)^{dg_2}, (X, X_y)^{(v+o)p_j}$$

from its oracle simulator. It chooses $\alpha \in Z_n$ randomly, and gives $A$ 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 $\alpha$.

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

$$(w^{y'}, g^{y'}, w' (X, X_y)^{r(i+d'+b)}g_2^{r_1}, (X, X_y)^{r_2})$$

When $A$ requests the challenge ciphertext for messages $M_0, M_1$, identity vector $(I_1', \cdots, I_l')$ and $T'$, $B$ 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 $t_0, t_1, \ldots, t_l \in Z_n$ chooses $\beta \in \{0, 1\}$ and creates the ciphertext as

$$(C = M_\beta \sigma(g^{y'}g_2^{y'}, g), C_0 = g^{y'}g_2^{y'}, \{C_{i,1}, C_{i,2}, C_{i,3}\}^I_{i=0})$$

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

$$(w^{y'}g_2^{y'}, v^{y'}g_2^{y'}, (u^{y'}h^{y'})^{d'v_{i+b}}g_2^{y'}g_2^{y'}, (w^{y'}h)^{d'v_{i+b}}g_2^{y'}, g^{y'v_{i+b}}),$$

else the element group is

$$(w^{y'}g_2^{y'}, v^{y'}g_2^{y'}, (u^{y'}h)^{d'v_{i+b}}g_2^{y'}, g^{y'v_{i+b}});$$

2. $i = k$: The ciphertext-element-group is $(T_1, T_3, T_5) = (w^{y'}g_2^{y'}T, d^{v_{i+b}}g_2^{y'})$.
3. $i > k$: The ciphertext-element-group is $(w'v^i, (u''h)^i, g^i)$.

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

If $T \in G_{p_3}$, then we must argue that $al + b$ and $al_i + b$ both appear to be uniformly random modulo $p_3$; this follows from pairwise independence of the function $al + b$ modulo $p_3$, since we have restricted the Type-1 adversary to choose $I$ and $I_i$ so that $I \neq I_i$ modulo $p_3$. This means that the $G_p$ 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, $O$ has produced a properly distributed EST-3-k-CT which means that $O$ has properly simulated the responses of $O_i$ and $B$ is playing Game $S_{k,2}$.

Particularly, we need overcome the paradox in the game hopping from Game $S_{k-1,1}$ to Game $S_{k,2}$ 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 way.)

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

Thus, Under Assumptions 3, no PPT attacker can distinguish between $O_i$ and $O_i$ 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 $O_i$ and $O_i$ 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 $B$ interacts with one of $O_i$, $O_i$. $O$ receives $g, g, X, Y, T$. $O$ will simulate either $O_i$ or $O_i'$ with $B$, depending on the value of $T$ (which is either in $G$ or $G_{p_3}$). $O$ picks values $a, b, c, d, a_0, b_0, c_0, d_0 \in Z_p$ uniformly at random and sets $u = g^a, h = g^b, v = g^c, w = g^d, u_0 = g^{a_0}, h_0 = g^{b_0}, v_0 = g^{c_0}, w_0 = g^{d_0}$. $B$ initially obtains the group elements

$$g, u, v, w, X, Y, w' = (Y Y^3)^0, g' = (Y Y^3)^0, v' = (Y Y^3)^0,$$

$$u_0, v_0, w_0, w_0' = (Y Y^3)^0, g_0' = (Y Y^3)^0, v_0' = (Y Y^3)^0$$

(85)

from its oracle simulator where $Z, Y_0, \psi \in Z_p$ are randomly chosen. These are properly distributed, with $g' = X_1$ and $g_0' = X_2$. Note that this sets $a_1$ equal to $c$ modulo $p_2$ and $p_3$, and $a_2$ equal to $c_0$ modulo $p_2$ and $p_3$. It chooses $\alpha \in Z_p$ randomly, and gives $A$ the following public parameters in Eq 83. We note that $B$ knows the master secret key $\alpha$. When $A$ requests a normal update key or a normal decryption key, $B$ can responds by using the usual key generation algorithm, since it knows $\alpha$. 

PLOS ONE | https://doi.org/10.1371/journal.pone.0195204 April 12, 2018 51 / 76
When $\mathcal{A}$ makes a secret key query for the identity $ID_0 = (I_1, \ldots, I_t)$, then $\mathcal{B}$ makes its challenge HIBE-key-type query for $I_0$. $\mathcal{O}$ responds as follows. It chooses $y', r, r_1, r_2 \in \mathbb{Z}_N$ randomly and responds with:

$$(w', g', w'(u^t h)'(Y_2 Y_3)^v, g'(Y_2 Y_3)^v)$$

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 $(I_1', \ldots, I_t')$ 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 $t_i', t_i^1, \ldots, t_i^k, t_i^k, \ldots, t_i \in \mathbb{Z}_p^*$, chooses $b \in \{0, 1\}$ and creates the ciphertext as

$$(C = M_i e(X_t X_2, g)^a, C_0 = X_t X_2, \{C_{i,1}, C_{i,2}, C_{i,3}\}_{i=0}^k)$$

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

$$((X_t X_2)^d(X_t X_2)^{a_0'}, (X_t X_2)^{v_{d_i}(a_0 T + b_0)}, (X_t X_2)^{v_{d_i}})$$

else the ciphertext-element-group is

$$((X_t X_2)^d(X_t X_2)^{a'}, (X_t X_2)^{v_{d_i}(a' + b)}, (X_t X_2)^{v_{d_i}})$$

This sets $X_t^d = g_{X_t^1}$ and $X_2^d = g_{X_2^1}$, which is uniformly random because the value of $d$ and $d_0$ modulo $p_2$ will not appear elsewhere. It implicitly sets $g^h = X_{t_1}^1$. This is identically distributed to a response from $O_2$, with $a', b'$ equal to $a$, $b$ modulo $p_2$, and $a_0 = c$ modulo $p_2$, $a_2 = c_0$ 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. $i = 0$: If $i = 0$, the ciphertext-element-group is

$$(T_1, T_3, T_2) = ((X_t X_2)^d T_{a_0}, \{a_0 T + b_0\}, T);$$

If $k > 0$, the ciphertext-element-group is

$$(T_1, T_3, T_2) = ((X_t X_2)^d T, \{a' + b\}, T);$$

3. $i > k$: The ciphertext-element-group is

$$((X_t X_2)^d v_{d_i}, (u^t h)^v, g^v)$$

If $T \in G_{p_2}$, then the response for ciphertext-type query $i$ is identically distributed to a response from $O'_i$.

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

If $T \in G$, then this response additionally has terms in $G_{p_2}$ which are appropriately distributed with $c = a_1$, $a = a'$, $b = b'$ modulo $p_2$ or $c_0 = a_2, a_0 = a'$, $b_0 = b'$ 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-4b-CT and $\mathcal{B}$ is playing Game $S_k$.

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

Thus, Under Assumptions 4, no PPT attacker can distinguish between $O'_k$ and $O'_k\mathcal{O}$ with non-negligible advantage. Thus, no PPT attacker can distinguish between $S_k$ and $S_k'$ with non-negligible advantage.
Lemma 5 Under Assumptions 3, no PPT attacker can distinguish between $O'_k$ and $O'_t$ 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 $B$ interacts with one of $O'_k, O'_t$. $O$ receives $g, g_2, X_1, X_3$. $T, O$ will simulate either $O'_k$ or $O'_t$ with $B$, depending on the value of $T$ (which is either in $G_p$, or $G_{p_3, p_3}$). $B$ initially obtains the group elements in Eq 82

$$g, u, h, v, w, g^i, g_{2i}, (X_i, X_3)^i g_{2i}, (X_i, X_3)^i g_{2i}, (X_i, X_3)^i g_{2i},$$

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

When $A$ makes a secret key query for the identity $ID_j = (I_1, \ldots, I_j)$, then $B$ makes its challenge HIBE-key-type query for $I_j$, $O$ responds as follows. It chooses $y', r, r_1, r_2 \in Z_n$ randomly and responds with:

$$(w', g', v' (X_i, X_3)^i g_{2i}, (X_i, X_3)^i g_{2i}, (X_i, X_3)^i g_{2i})$$

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

When $A$ requests the challenge ciphertext for messages $M_0, M_1$, identity vector $(I_1', \ldots, I_j')$ and $T'$, $B$ makes a ciphertext-type query to the oracle for each $I_j'$ and $T'$. In response to each query for $I_j'$ or $T'$, $O$ gets random $t_0, t_1, \ldots, t_i \in Z_{p'}$ chooses $\beta \in \{0, 1\}$ and creates the ciphertext as

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

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

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

   $$(w, g_{2i}^i, v_{2i}^i, (u_i h_i)^{g_{2i}^i}, (u_i h_i)^{g_{2i}^i}, (u_i h_i)^{g_{2i}^i}, (u_i h_i)^{g_{2i}^i})$$

   else the element group is

   $$(w, g_{2i}^i, v_{2i}^i, (u_i h_i)^{g_{2i}^i}, (u_i h_i)^{g_{2i}^i}, (u_i h_i)^{g_{2i}^i}, (u_i h_i)^{g_{2i}^i})$$. This is identically distributed to a response from $O_2$.

2. $i = k$: $O$ chooses $z \in Z_n$ randomly and responds with the ciphertext-element-group

   $$(T_1, T_1, T_2) = (w, g_{2i}^i, v_{2i}^i, (u_i h_i)^{g_{2i}^i}, (u_i h_i)^{g_{2i}^i}, (u_i h_i)^{g_{2i}^i}, (u_i h_i)^{g_{2i}^i})$$

   if $k > 0$. Else if $k = 0$, $O$ responds with

   $$(w, g_{2i}^i, v_{2i}^i, (u_i h_i)^{g_{2i}^i}, (u_i h_i)^{g_{2i}^i}, (u_i h_i)^{g_{2i}^i}, (u_i h_i)^{g_{2i}^i})$$. We note that the $G_{p_3}$ parts here are properly distributed, since $c_1 = c$ modulo $p_2$ and $c_2 = c_0$ modulo $p_2$.

3. $i > k$: The ciphertext-element-group is $(w, g_{2i}^i, v_{2i}^i, (u_i h_i)^{g_{2i}^i}, (u_i h_i)^{g_{2i}^i}, (u_i h_i)^{g_{2i}^i})$. This is identically distributed to a response from $O_1$.

When $T \in G_p$, 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, $O$ has properly simulated the responses of $O'_k$ and this will be a properly distributed EST-2^k-CT and so $B$ is playing Game $S_{k,1}$.

When $T \in G_{p_3, p_3}$, we must argue that the values $al + b$ and $al + b$ appear uniformly random modulo $p_3$: this follows by pairwise independence of $al + b$ 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, $O$ has produced a properly distributed
EST-4^4-CT and $\mathcal{O}$ has properly simulated the response of $O_i^a$ in this case. So $B$ is playing Game $S_{k,3}$. We have thus shown that $\mathcal{O}$ can use the output of $B$ 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 $O_i^a$ and $O_i^c$ with non-negligible advantage. It means $\mathcal{O}$ can use the output of $B$ to achieve a non-negligible advantage against Assumption 3.

Thus, Under Assumptions 3, no PPT attacker can distinguish between $O_i^a$ and $O_i^c$ 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 $E_{h_{0,1}}$ and $E_{h_{0,1}}$ with non-negligible advantage.

**Proof** We assume $B$ 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 $B$, depending on the value of $T$ (which is either in $G_1$ or $G_{p_1, p_3}$). $\mathcal{O}$ picks values $a, b, c, d, a_0, b_0, c_0, d_0 \in Z_N$ uniformly at random and sets $u = g^a$, $h = g^b$, $v = g^c$, $w = g^d$, $u_0 = g^{a_0}$, $h_0 = g^{b_0}$, $v_0 = g^{c_0}$, $w_0 = g^{d_0}$. $B$ initially obtains the group elements in $\text{Eq 82}$

\[
\begin{align*}
g, u, h, v, w, g^i g_i^2, & \ (X_i X_i)^{\lambda g_i^2}, (X_i X_i) g_i^{2\lambda}, (X_i X_i)^{g_i^2}, (X_i X_i)^{g_i^2} g_i^{2\lambda}, \\
u_0, h_0, v_0, w_0, & \ (X_i X_i)^{\lambda g_i^2}, (X_i X_i) g_i^{2\lambda}, (X_i X_i) g_i^{2\lambda}, (X_i X_i)^{g_i^2} g_i^{2\lambda}
\end{align*}
\]

from its oracle simulator who additionally chooses $s, \gamma, \delta_1, \delta_2, y_1, y_2, \in Z_n$ randomly.

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

When $A$ requests the challenge ciphertext for messages $M_0, M_1$, identity vector $(\theta_1, \ldots, \theta_j)$ and $T$, $B$ makes a ciphertext-type query to the oracle for each $I_i^G$ and $T$. When $B$ makes a ciphertext-type query for some identity $I$, $O$ responds by choosing a random $t \in Z_N$ and returning $(w^t g_i^{cg_i^2} g_i^{2\lambda}, g^{i^2 \lambda} g_i^{2\lambda}, (u^t h)^{i^2 g_i^{2\lambda}} g_i^{g_i^2})$ to $B$ as same as $\text{Eq 10}$. When $B$ makes a ciphertext-type query for some time $T$, $O$ responds by choosing a random $t_0 \in Z_N$, and returning $(w^t g_i^{cg_i^2} g_i^{2\lambda}, g^{i^2 \lambda} g_i^{2\lambda}, (u^t h)^{i h^t g_i^{2\lambda}} g_i^{g_i^2})$ as same as $\text{Eq 11}$ to $B$. Then $B$ creates the ESF-1 ciphertexts successfully.

When $A$ requests the secret key of an identity vector $ID_j = (I_1, \ldots, I_j)$, $B$ creates the ESF-2-SK key by the HIBE-type query response from $O$ and the secret key for $ID_j$ in some node $\theta$ is

\[
\begin{align*}
S_{i,0}^\theta = g^\lambda, S_{i,1} = g^\mu, S_{i,2} = g^\nu, S_{i,3} = (u^b h)^{i \lambda} g^{\nu}, i \in \{1, \ldots, j - 1\} \\
S_{i,0}^\theta = g^\nu \sum_{i=1}^{j-1} w^\mu, S_{i,1} = g^\mu, S_{i,2} = v^\nu (X_i X_i)^{\lambda (\mu + b)} \lambda g_i^{2\lambda}, S_{i,3} = (X_i X_i)^{\nu \lambda} g_i^{2\lambda}
\end{align*}
\]

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

When $B$ creates the IBE private key with the index pair $(h, i)$ for some time $T$ for the identity vector $(I_1, \ldots, I_j)$ in the index $h$ node, the update key with an index pair $(h, i)$ is generated as follows:
1. \( i_c < h_1 \): It randomly chooses \( y_0, \ldots, y_{j-1}, \lambda_1, \ldots, \lambda_{j-1}, r_0, r_1, \ldots, r_{j-1}, z, z' \in \mathbb{Z}_n \) and generates a ESF-2-UK TUK\(_{D_{lb}, T, 0,b}\):

\[
U_{0,0} = g^{|s|_n} \sum_{i=1}^{j-1} x_i \omega_i, U_{0,1} = g^{n_i}, U_{1,2} = v_0^{|a|_0}, U_{1,3} = (X_1 X_3)^{|(a+T+b)|_0} g_2^z,
\]

\[
U_{0,3} = (X_1 X_3)^{|a|_0} g_2^z,
\]

\[
U_{i,0} = g^{s_i \omega_i}, U_{i,1} = g^{n_i}, U_{i,2} = g^{n_i}, U_{i,3} = (v_i h)^{n_i} w_i, i \in \{1, \ldots, j-1\}
\]

It implicitly sets \( g^{n_i} \) to be \( X_1^{\gamma_i} \) and that is a properly distribution ESF-2-UK.

2. \( i_c = h_1 \): \( B \) chooses random values \( y_1, \ldots, y_{j-1}, \lambda_1, \ldots, \lambda_{j-1}, r_1, \ldots, r_{j-1} \in \mathbb{Z}_n \) and forms the challenge key as:

\[
U_{0,0} = g^{|s|_n} \sum_{i=1}^{j-1} x_i \omega_i, U_{0,1} = T_1, U_{0,2} = T_3, U_{0,3} = T_2
\]

\[
U_{i,0} = g^{s_i \omega_i}, U_{i,1} = g^{n_i}, U_{i,2} = g^{n_i}, U_{i,3} = (v_i h)^{n_i} w_i, i \in \{1, \ldots, j-1\}
\]

where \( (T_0, T_1, T_2, T_3) \) is the challenge IBE key queried to \( \mathcal{O} \) who chooses a random \( y_0 \in \mathbb{Z}_n \) and returns \( (T_0, T_1, T_2, T_3) = (w_0^{|b|_0}, g^{n_i}, v_0^{(b+T+h)} T^{n_i h}, T) \) to \( B \).

3. \( i_c > h_1 \): It simply generates a normal IBE private key.

In the challenge IBE key, it implicitly sets \( g^{n_i} \) 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 \( G_{p_2} \) terms here), and so this will be a properly distributed normal key and \( B \) is playing Game \( E_{b_{-1,2}} \). If \( T \in G_{p_1,p_2} \), then this matches the distribution of \( O_{b_{2}} \) (note that \( a, b \) modulo \( p_2 \) are uniformly random and do not occur elsewhere - so there are random \( G_{p_2} \) terms attached to the last two group elements) and then \( B \) is playing Game \( E_{b_{-1,2}} \).

Hence, if a PPT attacker can distinguish between \( E_{b_{-1,2}} \) and \( E_{b_{-1,4}} \) with non-negligible advantage, \( \mathcal{O} \) can distinguish between \( O_0 \) 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 \( O_0 \) and \( O_{5/2} \) with non-negligible advantage. Thus, no PPT attacker can distinguish between \( E_{b_{-1,2}} \) and \( E_{b_{-1,4}} \) 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 \( E_{b_{3,4}} \) and \( E_{b_{3,2}} \) with non-negligible advantage.

**Proof** We assume \( B \) interacts with one of \( O_{1/2}, \) and \( O_1 \). \( O \) receives \( g, g_3, X_1 X_2, Y_2 Y_3, T \). \( O \) will simulate either \( O_{1/2} \) or \( O_1 \) with \( B \), depending on the value of \( T \) (which is either in \( G \) or \( G_{p_1,p_2} \)). \( O \) picks values \( a, b, c, d, a_0, b_0, c_0, d_0 \in \mathbb{Z}_n \) uniformly at random and sets \( u = g^a, h = g^b, v = g^e, w = g^d, u_0 = g^{a_0}, h_0 = g^{b_0}, v_0 = g^{c_0}, w_0 = g^{d_0}. \) \( B \) initially obtains the group elements in Eq 84

\[
g, u, h, v, w, X_1 X_2, w^e (Y_2 Y_3)^v, g^e (Y_2 Y_3)^v, v^e (Y_2 Y_3)^u,
\]

\[
u_0, h_0, v_0, w_0, w_0^e (Y_2 Y_3)^v, g_0^e (Y_2 Y_3)^v, v_0^e (Y_2 Y_3)^u
\]

from its oracle simulator, and gives \( A \) the public parameters in Eq 83.

When \( A \) requests the challenge ciphertext for messages \( M_0, M_1 \), identity vector (\( I'_1, \ldots, I'_i \)) and \( T \), \( B \) makes a ciphertext-type query to the oracle for each \( I'_i \) and \( T \). When \( B \) makes a ciphertext-type query for some identity \( I' \), \( O \) responds by choosing a random \( t \in \mathbb{Z}_n \) and
where \( y_0 \) is the challenge IBE key queried to \( O \) and returns \((w_0^0, v_0^0, g_{i1}^{(a_i+b)} g_{i2}^{(a_i+b)})\) for a time \( T \) in the index node in the binary tree \( BT_{ID_0} = (n_1, \cdots, b_{-1}) \), the update key with an index pair \((h, i)\) is generated as follows:

1. \( i < h \): \( B \) chooses random values \( y_1, \cdots, y_{j-1}, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_{j-1}, z, z' \in Z_n \) and generates a ESF-2 TUK\(_{IBE,h}\):

\[
S_{0,0} = g^{y_1} w_0^{0}, \quad S_{1,0} = g^{\lambda_1}, S_{2,0} = (u^t h)^{y_0} g_{i1}^{z'}, i \in \{1, \cdots, j - 1\}
\]

\[
S_{0,0} = g^{y_1} w_0^{0}, \quad S_{1,0} = g^{\lambda_1}, S_{2,0} = (u^t h)^{y_0} g_{i1}^{z'}, i \in \{1, \cdots, j - 1\}
\]

2. \( i = h \): \( B \) chooses random values \( y_1, \cdots, y_{j-1}, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_{j-1} \in Z_n \). \( B \) forms the challenge key as:

\[
U_{0,0} = g^{y_1} \sum_{i=1}^{j-1} w_0^{0}, \quad U_{0,1} = g^{\lambda_1}, U_{0,2} = (u^t h)^{y_0} (y_1 Y_1)^{r_1} Z_1, U_{0,3} = g^{\lambda_1} U_{1,2} = g^{y_1}, U_{1,3} = (u^t h)^{y_0} v_0^{i}, i \in \{1, \cdots, j - 1\}
\]

where \((T_0, T_1, T_2, T_3)\) is the challenge IBE key queried to \( O \) who chooses a random \( y_0 \in Z_n \) and returns \((w_0^0, g_{i1}^{\lambda_1}, v_0^{i} T^{a_i+b}, T)\) to \( B \).

3. \( i > h \): It simply generates a normal HIBE private key.

As in the previous lemma, this implicitly sets \( g_{i1} \) 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 \( T \in G_{p_2, p_3} \), these last two terms will have random elements of \( G_p \) attached (matching the distribution of \( O_{5/2} \)) and then \( B \) is playing Game \( E_{h,1} \). And when \( T \in G_3 \), these last two terms will have random elements in both \( G_p \) and \( G_p \) attached (matching the distribution of \( O_3 \)) and then \( B \) is playing Game \( E_{h,2} \).

Hence, if a PPT attacker can distinguish between \( E_{h,1} \) and \( E_{h,2} \) with non-negligible advantage, \( O \) can distinguish between \( O_{5/2} \) and \( O_3 \) with non-negligible advantage. It means \( O \) 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 \( E_{h,1} \) and \( E_{h,2} \) with non-negligible advantage.
Lemma 8 Under Assumptions 3, no PPT attacker can distinguish between $O_3$ and $O_{3,3}$ with non-negligible advantage. So no PPT attacker can distinguish between $F_{h-1}$ and $F_h$ with non-negligible advantage.

Proof We assume $B$ interacts with one of $O_3$, $O_{3,3}$. $O$ receives $g, g_2, X_1, X_3, Y_1, Y_3, T$. $O$ will simulate either $O_3$ or $O_{1/2}$ with $B$, depending on the value of $T$ (which is either in $G_{P_1}$ or $G_{P_{1,3}}$).

$O$ picks values $a, b, c, d, a_0, b_0, c_0, d_0 \in \mathbb{Z}_n$ uniformly at random and sets $u = g^a, h = g^b, v = g^c, w = g^d, u_0 = g^{a_0}, h_0 = g^{b_0}, v_0 = g^{c_0}, w_0 = g^{d_0}$. $B$ initially obtains the group elements in Eq 82

$$g, u, h, v, w, g_2^u, (X_1, X_3)^u g_2^v, (X_1, X_3)^v g_2^w$$

$$u_0, h_0, v_0, w_0, (X_1, X_3)^u g_2^v, (X_1, X_3)^v g_2^w$$

from its oracle simulator who additionally chooses $s, \gamma, \delta, \alpha, \beta, \gamma_2, \delta_2, \epsilon_2, \delta_1, \gamma_1 \in \mathbb{Z}_N$ randomly. $B$ chooses $\alpha \in \mathbb{Z}_N$ randomly, and gives $A$ the following public parameters in Eq 83. We note that $B$ knows the master secret key $\alpha$.

When $A$ requests the challenge ciphertext for messages $M_0, M_1$, identity vector $(I_1', \ldots, I_j')$ and $T'$, $B$ makes a ciphertext-type query to the oracle for each $I_i'$ and $T'$. When $B$ makes a ciphertext-type query for some identity $I_i$, $O$ responds by choosing a random $t \in \mathbb{Z}_n$ and returning $(w_c^{I_i} v^{I_i} g_2^{v}, g_2^{v}, \alpha^t g_2^{v T'})$ to $B$. When $B$ makes a ciphertext-type query for some time $T'$, $O$ responds by choosing a random $t_0 \in \mathbb{Z}_n$ and returning $(w_c^{I_i} v^{I_i} g_2^{v}, g_2^{v}, \alpha^t g_2^{v T'})$ to $B$. Then $B$ creates the ESF-1 ciphertexts successfully.

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

$$((g)^{r_1}, (g)^{r_2}, (g)^{r'}, (X_1, X_3)^{r T+h} g_2^v, (X_1, X_3)^{r T'} g_2^v)$$

to $B$. And then $B$ creates the ESF-2 update key by using the group elements.

When $B$ creates the HIBE private key with the index pair $(h, i)$ in the index node, the HIBE private key with an index pair $(h, i)$ is generated as follows:

1. $i < h$: It randomly chooses $y_1, \ldots, y_{j-1}, y', \lambda_1, \ldots, \lambda_{j-1}, r_1, \ldots, r_{j-1}, r', y', z, z' \in \mathbb{Z}_n$ and generates an ESF-3-SK $PSK_{HIBE,k'}$,

$$K_{ib} = g^{h w}, K_{i1} = g^{h}, K_{i2} = g^{r}, K_{i3} = (u^{h})^{i} g^{w}, i \in \{1, \ldots, j-1\}$$

$$K_{i0} = g^{y_1 - \sum_{i=1}^{j-1} y_i (X_1, X_3)^{y_i}}, K_{i1} = (X_1, X_3)^y g^{y_1},$$

$$K_{i2} = (X_1, X_3)^{y'} (X_1, X_3)^{y (r_1+1)} g_2^v, K_{i3} = (X_1, X_3)^{y'} g^{y_1}$$

It implicitly sets $g^{y_0}$ to be $X_j^{y_1}$ and $g^{y'}$ to be $X_j^{y_1'}$ and that is a properly distribution ESF-3-SK.

2. $i = h$: $B$ chooses random values $y_1, \ldots, y_{j-1}, \lambda_1, \ldots, \lambda_{j-1}, r_1, \ldots, r_{j-1} \in \mathbb{Z}_n$. $B$ forms the challenge key as:

$$K_{ib} = g^{h w}, K_{i1} = g^{h}, K_{i2} = g^{r}, K_{i3} = (u^{h})^{i} g^{w}, i \in \{1, \ldots, j-1\}$$

$$K_{i0} = g^{y_1 - \sum_{i=1}^{j-1} y_i T_0}, K_{i1} = T_1, K_{i2} = T_2, K_{i3} = T_2$$

where $(T_0, T_1, T_2, T_3)$ is the challenge HIBE key queried to $O$ who chooses a random $r, r_1, r_2 \in \mathbb{Z}_N$ and returns $(T_0, T_1, T_2, T_3) = (T^{y}, T, T^v (X_1, X_3)^{y (r_1+1)} g_2^v, (X_1, X_3)^{y'} g_2^v)$ to $B$. 

PLOS ONE | https://doi.org/10.1371/journal.pone.0195204 April 12, 2018 57 / 76
3. $i_c > h$: It randomly chooses $y_1, \ldots, y_{j-1}, r_1, \ldots, r_{j-1}, r'_j, z, z' \in \mathbb{Z}_n$ and generates an ESF-2-SK $PSK_{HIBE,h}$.

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

$$K_{j,0} = g^{y_j} \sum_{i=1}^{j-1} x_i w^{y_i}, K_{j,1} = g^{y_j}, K_{j,2} = g^{r_j} (X_{X_j})^{(a_j + b_j)} g_2, K_{j,3} = (X_{X_j})^{g_2} g_2$$

It implicitly sets $g^{y_j}$ to be $X_{X_j}^i$ and that is a properly distribution ESF-2-SK.

In the challenge HIBE key, it implicitly sets $g^{y_j}$ to be the $G_{p_j}$ part of $T$. If $T \in G_{p_j}$, then this matches the distribution of $O_{3}$ (since there are no $G_{p_j}$ terms here), and so this will be a properly distributed normal key and $B$ is playing Game $F_{h,-1}$. If $T \in G_{p_j}$, 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 $G_{p_j}$ terms attached to the last two group elements) and then $B$ is playing Game $F_{h}$.

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

Thus, Under Assumptions 3, no PPT attacker can distinguish between $O_{3}$ and $O_{3,1}$. Hence, if a PPT attacker can distinguish between $F_{h,-1}$ and $F_{h}$ with non-negligible advantage. Thus, no PPT attacker can distinguish between $F_{h,-1}$ and $F_{h}$ 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 $F_{h,-1}$ and $F_{h}$, 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 $I \in Z_n$, $O$ chooses $r_1, r_2, r', y' \in \mathbb{Z}_n$ randomly and returns the group elements

$$((X_i X_j)^{y_1}, (X_i X_j)^{y_2}, (X_i X_j)^{y'} (X_i X_j)^{(a_j+b_j)} g_2, (X_i X_j)^{y'} g_2)$$

to $B$. And then $B$ creates the ESF-3 update key by using the group elements.

When $B$ creates the IBE private key with the index pair $(h, i_c)$ for some identity vector $(I_1, \ldots, I_{j-1})$ 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. $i_c < h$: It randomly chooses $y_1, \ldots, y_{j-1}, y_0, \lambda_1, \ldots, \lambda_{j-1}, r_1, \ldots, r_{j-1}, r'_0, z, z' \in \mathbb{Z}_n$ and generates an ESF-3-UK $EUK_{HIBE,h}$.

$$U_{0,0} = g^{x_0 w + \sum_{i=1}^{j-1} x_i} (X_i X_j)^{y_0}, U_{0,1} = (X_i X_j)^{y_0},$$

$$U_{0,2} = (X_i X_j)^{(a_0+b_0)} (X_i X_j)^{g_2}, U_{0,3} = (X_i X_j)^{y_0} g_2$$

$$U_{0} = g^{x_0 w}, U_{r_i} = g^{y_i}, U_{i,2} = g^{y_i}, U_{i,3} = (u^i h)^{y_i} g_2, i \in \{1, \ldots, j - 1\}$$

It implicitly sets $g^{y_0}$ to be $X_{X_j}^i$ and $g^{y_i}$ to be $X_{X_j}^i$ and that is a properly distribution ESF-3-UK.
2. \( i_c = h; B \) chooses random values \( y_1, \ldots, y_{j-1}, \lambda_1, \ldots, \lambda_{j-1}, r_1, \ldots, r_{j-1} \in \mathbb{Z}_p \). \( B \) forms the challenge key as:

\[
U_{i,0} = g^{x_h} \sum_{i=1}^{j-1} h_i T_{i,0},
\]

\[
U_{i,0} = T_1, U_{i,2} = T_3, U_{i,3} = T_2
\]

\[
U_{i,0} = g^{h_i} U_{i,1} = g^{\sigma_i}, U_{i,2} = g^{\sigma_i}, U_{i,3} = (u_i h)^{\sigma_i}, i \in \{1, \ldots, j-1\}
\]

where \((T_0, T_1, T_2, T_3)\) is the challenge IBE key queried to \( O \) who chooses a random \( r, r_1, r_2 \in \mathbb{Z}_p \) and returns \((T_0, T_1, T_2, T_3) = (T h_i, T, T^0 (X, X_3)^{\sigma_i h_i/2}, (X, X_3)^{g_i z_i})\) to \( B \).

3. \( i_c > h \); it randomly chooses \( y_1, \ldots, y_{j}, \lambda_1, \ldots, \lambda_{j-1}, r_1, \ldots, r_{j-1}, r'_1, z, z' \in \mathbb{Z}_p \) and generates an ESF-2-UK IBE key.

\[
U_{i,0} = g^{h_i} U_{i,1} = g^{\sigma_i}, U_{i,2} = g^{\sigma_i}, U_{i,3} = (u_i h)^{\sigma_i}, i \in \{1, \ldots, j-1\}
\]

It implicitly sets \( g^{\sigma_i} \) to be \( X_1^{i-h_i} \) and that is a properly distributed ESF-2-UK.

In the challenge IBE key, it implicitly sets \( g^{\sigma_i} \) to be \( G_{p_i} \) part of \( T \). If \( T \in G_{p_i} \), then this matches the distribution of \( O_{j,1} \) (since there are no \( G_{p_i} \) terms here), and so this will be a properly distributed normal key and \( B \) is playing Game \( F_{1,h-1} \). If \( T \in G_{p_i,p_j} \) then this matches the distribution of \( O_{j,2} \) (note that \( a, b \) modulo \( p_j \) are uniformly random and do not occur elsewhere- so there are random \( G_{p_i} \) terms attached to the last two group elements) and then \( B \) is playing Game \( F_{1,h} \).

Hence, if a PPT attacker can distinguish between \( F_{1,h-1} \) and \( F_{1,h} \) with non-negligible advantage, \( O \) can distinguish between \( O_{j,1} \) and \( O_{j,2} \) with non-negligible advantage. It means \( O \) can gain a non-negligible advantage against Assumption 3.

Thus, Under Assumptions 3, no PPT attacker can distinguish between \( O_{j,1} \) and \( O_{j,2} \) with non-negligible advantage. Thus, no PPT attacker can distinguish between \( F_{1,h-1} \) and \( F_{1,h} \) with non-negligible advantage.

**Lemma 10** Under Assumptions 4, no PPT attacker can distinguish between \( O_{j,2} \) and \( O_{j,3} \) 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 \( B \) interacts with one of \( O_{j,2}, O_{j,3} \). \( O \) receives \( g, g', X, X_3, Y_2, Y_3, T \). \( O \) will simulate either \( O_{j,2} \) or \( O_{j,3} \) with \( B \), depending on the value of \( T \) (which is either in \( G \) or \( G_{p_i,p_j} \)). \( O \) picks values \( a, b, c, d, a_0, b_0, c_0, d_0 \in \mathbb{Z}_p \) 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 \( \sigma_1, \sigma_2, y', t_3, z \in \mathbb{Z}_p \) and then \( B \) initially obtains the group elements

\[
g, u, h, v, T, w^i (Y_2 Y_3)^{1/d}, g^{i} (Y_2 Y_3), \; w^i (Y_2 Y_3)^{1/d},
\]

\[
u_0, h_0, v_0, w_0, w_0 (Y_2 Y_3)^{1/d}, g^{i} (Y_2 Y_3), \; v_0 (Y_2 Y_3)^{1/d},
\]

\[(86)\]

We note that this sets \( \psi_1 = d \) modulo \( p_j \) and \( p_i \). It implicitly sets \( g^{i} \) to be the \( G_{p_i} \) part of \( T \). If \( T \in G_{p_i,p_j} \), this is distributed identically to the initial elements provided by \( O_{j,2} \). If \( T \in G \), this is distributed identically to the initial elements provided by \( O_{j,3} \).
Upon receiving a challenge IBE-key-type query for \( T \in \mathbb{Z}_m \), \( \mathcal{O} \) chooses \( r_1, r_2, r' \in \mathbb{Z}_n \) randomly and returns the group elements
\[
((g)^{r_1}, (g)^{r_2}, (g)^{r'}(X_1X_5)\xi g_{v_1}^{a_0}g_{k_1}^{r_1} g_{z_1}^r, (X_1X_5)^{r'}g_{z_1}^r)
\]
to \( B \). And then \( B \) creates the ESF-2 update key by using the group elements.

When \( B \) creates the HIBE private key with the index pair \((h, i_3)\) for some identity vector \((I_1, \cdots, I_l)\) in the index \( h \) node, the HIBE private key with an index pair \((h, i_3)\) is generated as follows:

When \( A \) requests the challenge ciphertext for messages \( M_{10}, M_{11}, \) identity vector \((I'_1, \cdots, I'_l)\) and \( T' \), \( B \) makes a ciphertext-type query to the oracle for each \( I'_i \) and \( T' \). When \( B \) makes a ciphertext-type query for some identity \( I' \), \( \mathcal{O} \) responds by choosing a random \( a \in \mathbb{Z}_N \) and returning \((w'g_{z_1}^{y_0}g_{v_1}^{a_0}g_{k_1}^{r_1} g_{z_1}^{v_1}, (u' h)^{a_0}g_{k_1}^{r_1} g_{z_1}^{v_1}) \) to \( B \). When \( B \) makes a ciphertext-type query for some time \( T' \), \( \mathcal{O} \) responds by choosing a random \( t_0 \in \mathbb{Z}_N \) and returning
\[
(w'g_{z_1}^{y_0}g_{v_1}^{a_0}g_{k_1}^{r_1} g_{z_1}^{v_1}, (u' h_0)^{a_0}g_{k_1}^{r_1} g_{z_1}^{v_1})
\]
to \( B \). Then \( B \) creates 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 \( B \) interacts with one of \( O_{3,3}, O_{3,4} \). \( \mathcal{O} \) receives \( g, g_3, X_1X_5, Y_2Y_3, T \). \( \mathcal{O} \) will simulate either \( O_{3,3} \) or \( O_{3,4} \) with \( B \), depending on the value of \( T \)(which is either in \( G \) or \( G_{p_1, p_3} \)). \( \mathcal{O} \) picks values \( u, v, w, x, a_0, b_0, c_0 \in \mathbb{Z}_N \) uniformly at random and sets \( u = g^x, v = g^w, w = g^v, u_0 = g^y, h_0 = g^h, v_0 = g^v, w_0 = g^w \). \( B \) initially obtains the group elements
\[
, u, h, v, w, (Y_2Y_3)^x, (Y_2Y_3)^x, (Y_2Y_3)^x, (Y_2Y_3)^x, (Y_2Y_3)^x
\]
from its oracle simulator who additionally chooses \( s, y, y', z \in \mathbb{Z}_N \) randomly. We note that this is properly distributed and set \( \psi_1 = s \) 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 \( A \) requests the challenge ciphertext for messages \( M_{10}, M_{11}, \) identity vector \((I'_1, \cdots, I'_l)\) and \( T' \), \( B \) makes a ciphertext-type query to the oracle for each \( I'_i \) and \( T' \). When \( B \) makes a ciphertext-type query for some identity \( I' \), \( \mathcal{O} \) responds by choosing a random \( t \in \mathbb{Z}_N \) and returning
\[
(w'_g g_{v_1}^{u_0}, (X_1X_5)^{\zeta} g_{z_1}^x, (X_1X_5)^{\zeta} g_{z_1}^x, (X_1X_5)^{\zeta} g_{z_1}^x)
\]
to \( B \). When \( B \) makes a ciphertext-type query for some time \( T' \), \( \mathcal{O} \) responds by choosing a random \( t_0 \in \mathbb{Z}_N \) and returning
\[
(w'_g g_{v_1}^{u_0}, (X_1X_5)^{\zeta} g_{z_1}^x, (X_1X_5)^{\zeta} g_{z_1}^x)
\]
to \( B \). This implicitly sets \( g^t = X_1X_5 \). It also sets \( a' = a \) and \( b' = b \) modulo \( p_2 \). which are properly distributed because \( a,b \) modulo \( p_2 \) do not appear otherwise. Then \( B \) creates the ESF-5 ciphertexts successfully.

Upon receiving a challenge IBE-key-type query for \( T \in \mathbb{Z}_m \), \( \mathcal{O} \) chooses \( r, r', z \in \mathbb{Z}_n \) randomly and returns the group elements
\[
(w'_g g_{v_1}^{u_0}, g^t g_{z_1}^x, g' (Y_2Y_3), (Y_2Y_3)^x)
\]
to \( B \). And then \( B \) creates the ESF-3 update key by using the group elements.

When \( B \) creates the HIBE private key with the index pair \((h, i_3)\) for some identity vector \((I_1, \cdots, I_l)\) in the index \( h \) node, the HIBE private key with an index pair \((h, i_3)\) is generated as follows:
1. \( i_c < h_2 \): It randomly chooses \( y_1, \ldots, y_j, \lambda_1, \ldots, \lambda_{j-1}, r_1, \ldots, r_j, z, z' \in \mathbb{Z}_n \) and generates a ESF-4-SK \( \text{PSK}_\text{HIBE}_{\lambda_2} \)

\[
K_{i,0} = g^{y_1}w^{\lambda_1}, K_{i,1} = g^{y_2}, K_{i,2} = g^{y_3}, K_{i,3} = (u^{i}h)^{y_1}w^{\lambda_1}, i \in \{1, \ldots, j-1\}
\]

\[
K_{j,0} = g^{y_1} \sum_{i=1}^{j-1} r_i w^{(Y_2 Y_3)^{y_i}}, K_{j,1} = g^{y_2}(Y_2 Y_3)^{y_1},
\]

\[
K_{j,2} = v^{y_1}(u^{i}h)^{y_1}(Y_2 Y_3)^{y_1}, K_{j,3} = g^{y_2}(Y_2 Y_3)^{y_2}
\]

That is a properly distribution ESF-4-SK.

2. \( i_c = h_2 \): \( B \) chooses random values \( y_1, \ldots, y_{j-1}, \lambda_1, \ldots, \lambda_{j-1}, r_1, \ldots, r_{j-1} \in \mathbb{Z}_n \). \( B \) forms the challenge key as:

\[
K_{i,0} = g^{y_1}w^{\lambda_1}, K_{i,1} = g^{y_2}, K_{i,2} = g^{y_3}, K_{i,3} = (u^{i}h)^{y_1}w^{\lambda_1}, i \in \{1, \ldots, j-1\}
\]

\[
K_{j,0} = g^{y_1} \sum_{i=1}^{j-1} r_i w^{(Y_2 Y_3)^{y_i}}, K_{j,1} = T_1, K_{j,2} = T_2, K_{j,3} = T_2
\]

where \( (T_0, T_1, T_2, T_3) \) is the challenge HIBE key queried to \( \mathcal{O} \) who chooses a random \( r, r_1, r_2 \in \mathbb{Z}_n \) and returns \( (T_0, T_1, T_2, T_3) = (T^i, T, T^j, (u^{i}h)^{y_1}(Y_2 Y_3)^{y_1}, g^{y_2}(Y_2 Y_3)^{y_2}) \) to \( B \).

3. \( i_c > h_2 \): It randomly chooses \( y_1, \ldots, y_j, \lambda_1, \ldots, \lambda_{j-1}, r_1, \ldots, r_j, z, z' \in \mathbb{Z}_n \) and generates a ESF-3-SK \( \text{PSK}_\text{HIBE}_{\lambda_2} \)

\[
K_{i,0} = g^{y_1}w^{\lambda_1}, K_{i,1} = g^{y_2}, K_{i,2} = g^{y_3}, K_{i,3} = (u^{i}h)^{y_1}w^{\lambda_1}, i \in \{1, \ldots, j-1\}
\]

\[
K_{j,0} = g^{y_1} \sum_{i=1}^{j-1} r_i w^{(Y_2 Y_3)^{y_i}}, K_{j,1} = g^{y_2},
\]

\[
K_{j,2} = v^{y_1}(u^{i}h)^{y_1}(Y_2 Y_3)^{y_1}, K_{j,3} = g^{y_2}(Y_2 Y_3)^{y_2}
\]

That is a properly distribution ESF-3-SK.

In the challenge HIBE key, it implicitly sets \( g^{y_2} \) to be the \( G_{p_3} \) part of \( T \). If \( T \in G_{p_3^3} \), then this matches the distribution of \( O_{3,3} \), and so this will be a properly distributed normal key and \( B \) is playing Game \( F3_{h_{j-1}} \). If \( T \in G \), then this matches the distribution of \( O_{3,4} \) and then \( B \) is playing Game \( F3_{h_j} \).

Hence, if a PPT attacker can distinguish between \( F3_{h_{j-1}} \) and \( F3_{h_j} \) 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 \( F3_{h_{j-1}} \) and \( F3_{h_j} \) 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 \( G_{\text{ESF}^{-4}} \) and \( G_{\text{ESF}^{-5}} \) 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 keys.

Upon receiving a challenge HIBE-key-type query for \( I \in \mathbb{Z}_n \), \( \mathcal{O} \) chooses \( r, r', z, z' \in \mathbb{Z}_n \) randomly and returns the group elements

\[
(w' (Y_2 Y_3)^{z'}, g'(Y_2 Y_3)^{z'}, g'(u' h)^{y_1}(Y_2 Y_3)^{z'})
\]

to \( B \). And then \( B \) creates the ESF-4 secret key by using the group elements.
When $B$ creates the IBE private key with the index pair $(h, i)$ for some time $T$ for the identity vector $(I_1, \ldots, I_{j-1})$ in the index $h$ node, the update key with an index pair $(h, i)$ is generated as follows:

1. $i < h$: It randomly chooses $y_1, \ldots, y_j, \lambda_1, \ldots, \lambda_{j-1}, r_0, \ldots, r_{j-1}, z, z' \in \mathbb{Z}_n$ and generates a ESF-4-UK TUK$_{IBE,h}$:

$$U_{0,0} = g^{y_1 - \gamma_0 \sum_{i=1}^{j-1} h_i} w_0^n \left( Y_2 Y_3 \right)^{y_j h_i}, U_{0,1} = g^{y_1} \left( Y_2 Y_3 \right)^{y_0}$$
$$U_{0,2} = v_0^n \left( u_0^r h_0 \right)^{y_1} \left( Y_2 Y_3 \right)^{r'}, U_{0,3} = g^{y_1} \left( Y_2 Y_3 \right)^{z'},$$

$$U_{i,0} = g^{y_i} w_i^r, U_{i,1} = g^{v_i}, U_{i,2} = g^{v_i}, U_{i,3} = (u^\iota h)^{\iota} w_i^r, i \in \{1, \ldots, j-1\}$$

That is a properly distributed ESF-4-UK.

2. $i = h$: $B$ chooses random values $y_1, \ldots, y_j, \lambda_1, \ldots, \lambda_{j-1}, r_0, \ldots, r_{j-1} \in \mathbb{Z}_n$ and forms the challenge key as:

$$U_{0,0} = g^{y_1 - \gamma_0 \sum_{i=1}^{j-1} h_i} T_0, U_{0,1} = T_1, U_{0,2} = T_2, U_{0,3} = T_3$$
$$U_{i,0} = g^{y_i} w_i^r, U_{i,1} = g^{v_i}, U_{i,2} = g^{v_i}, U_{i,3} = (u^\iota h)^{\iota} w_i^r, i \in \{1, \ldots, j-1\}$$

where $(T_0, T_1, T_2, T_3)$ is the challenge IBE key queried to $O$ who chooses a random $r, r_1, r_2 \in \mathbb{Z}_n$ and returns $(T_0, T_1, T_2, T_3) = (T_0^h, T, T^n \left( u_0^r h_0 \right)^{y_1} \left( Y_2 Y_3 \right)^{r'}, g^{y_i} \left( Y_2 Y_3 \right)^z)$ to $B$.

3. $i > h$: It randomly chooses $y_0, y_1, \ldots, y_{j-1}, \lambda_1, \ldots, \lambda_{j-1}, r_0, \ldots, r_{j-1}, z, z' \in \mathbb{Z}_n$ and generates a ESF-3-UK TUK$_{IBE,h}$.

$$U_{0,0} = g^{y_1 - \gamma_0 \sum_{i=1}^{j-1} h_i} g_0^r, U_{0,1} = g^{y_1} g_0^r, U_{0,2} = v_0^n \left( u_0^r h_0 \right)^{y_1} \left( Y_2 Y_3 \right)^{r'},$$

$$U_{3,0} = g^{y_1} \left( Y_2 Y_3 \right)^{z'},$$

$$U_{i,0} = g^{y_i} w_i^r, U_{i,1} = g^{v_i}, U_{i,2} = g^{v_i}, U_{i,3} = (u^\iota h)^{\iota} w_i^r, i \in \{1, \ldots, j-1\}$$

That is a properly distributed ESF-3-UK.

In the challenge IBE key, it implicitly sets $g^r$ to be the $G_p$ part of $T$. If $T \in G_{p,p^3}$, then this matches the distribution of $O_{3,4}$, and so this will be a properly distributed normal key and $B$ is playing Game $F_{4h_{j-1}}$. If $T \in G$, then this matches the distribution of $O_{3,5}$, and then $B$ is playing Game $F_{4h_i}$.

Hence, if a PPT attacker can distinguish between $F_{4h_{j-1}}$ and $F_{4h_i}$ with non-negligible advantage, $O$ can distinguish between $O_{3,4}$ and $O_{3,5}$ with non-negligible advantage. It means $O$ 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 $F_{4h_{j-1}}$ and $F_{4h_i}$ 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 $G_{3,5}$ and $G_{3,6}$ with non-negligible advantage.

**Proof** We assume $B$ interacts with one of $O_{3,5}, O_{3,6}$, $O$ receives $g, g_2, X_1, X_3, Y_2, Y_3, T$. $O$ will simulate either $O_{3,5}$ or $O_{3,6}$ with $B$, depending on the value of $T$ (which is either in $G$ or $G_{p,p^3}$). $O$ picks values $a, b, c, d, a_0, b_0, c_0, d_0 \in \mathbb{Z}_n$ uniformly at random and sets $u = g^a, h = g^b, v = g^c, \ldots$. 

PLOS ONE https://doi.org/10.1371/journal.pone.0195204 April 12, 2018 62 / 76
\[ w = g^{d}, u_0 = g^{h_0}, v_0 = g^{h_0}, w_0 = g^{h_0} \]

\[ B \text{ initially obtains the group elements }\]

\[ g, u, v, w, T, w(Y_2Y_3)^{d_0}, g^0(Y_2Y_3)^{\sigma_0}, \]

\[ u_0, v_0, w_0, w(Y_2Y_3)^{d_0}, g^0(Y_2Y_3)^{\sigma_0}, v_0(Y_2Y_3)^{e_0} \]

(88)

from its oracle simulator where \( z, y_0, y, \sigma_1, \sigma_2 \in \mathbb{Z}_p \) are randomly chosen. We note that this set \( \psi = d \mod p_2 \) and \( \sigma_1 \). If \( T \in G_{p_2} \), 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 \( \alpha \in \mathbb{Z}_p \) randomly, and gives \( A \) the following public parameters in Eq 83. We note that \( B \) knows the master secret key \( \alpha \). When \( A \) requests a normal update key or a normal decryption key, \( B \) can respond by using the usual key generation algorithm, since it knows \( \alpha \).

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

\[
((X, Xg_2)^{y'}, (X, Xg_2)^{y'})(u^ih)^{(Y_2Y_3)^{\alpha}}, g^{(Y_2Y_3)^{\alpha}})
\]

And then \( B \) creates the ESF-4 secret key by using the group elements.

Upon receiving a challenge IBE-key-type query for \( T \in \mathbb{Z}_N \), \( \mathcal{O} \) chooses \( y', r, r_1, r_2 \in \mathbb{Z}_N \) randomly and responds with:

\[
((X, Xg_2)^{r_2i}, (X, Xg_2)^{r_3i})(u^ih)^{(Y_2Y_3)^{\alpha}}, g^{(Y_2Y_3)^{\alpha}})
\]

to \( B \). And then \( B \) creates 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 \( (I_1, \ldots, I_l) \) and \( T \), \( B \) makes a ciphertext-type query to the oracle for each \( I_j \) and \( T \). In response to each query for \( I_j \) or \( T \), \( \mathcal{O} \) gets random \( t_0, t_1, \ldots, t_l \in \mathbb{Z}_p \) chooses \( \beta \in \{0, 1\} \) and creates the ciphertext as

\[ (C = M_\beta e(X, Xg)^{\alpha}, C_0 = X, Xg^{\alpha}; C_{i1}, C_{i2}, C_{i3}, i_1)_{i=0} \]

where the ciphertext-element-group \( (C_{i1}, C_{i2}, C_{i3}) \) is defined as follows if \( i > 0 \):

\[
(T^{d_1}T^{a_1}g_2^{\beta_1}, T^{e_1}g_2^{\beta_1}, T^{(a_1+b_1)}g_2^{(a_1+b_1)}, T^{X})
\]

and \( (C_{0,1}, C_{0,2}, C_{0,3}) \) is defined as follows

\[
(T^{d_0}T^{a_1}g_2^{\beta_1}, T^{e_1}g_2^{\beta_1}, T^{(a_1+b_1)}g_2^{(a_1+b_1)}, T^{X})
\]

We note that this is very similar to the way \( \mathcal{O} \) behaves in the proof of Lemma 12. The only difference is the \( g_2^{y'}, g_2^{y'}, g_2^{y'} \) terms which have been added to the challenge key. As in the proof of Lemma 12, we have that if \( T \in G \), the \( G_{p_1} \) components of the challenge ciphertext are properly distributed as in a response from \( O_{3.5} \), since the value of \( c \mod p_1 \) is not revealed by the challenge key-type response (it is hidden by the random term \( Y_3^e \)). 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 \( T \in G_{p_2} \), \( \mathcal{O} \) 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 \( B \) 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 $O_{7/2^2}$ with non-negligible advantage. So no PPT attacker can distinguish between $F_{6,1-2}$ and $F_{6,1}$ with non-negligible advantage.

**Proof** We assume $B$ interacts with one of $O_{3,6}$ and $O_{7/2^2}$. $O$ receives $g, g_0, X_1 X_2 Y_2 Y_3, T$. $O$ will simulate either $O_{3,6}$ and $O_{7/2^2}$ with $B$, depending on the value of $T$ (which is either in $G$ or $G_{B, p_3}$). $O$ picks values $a, b, c, d, a_0, b_0, c_0, d_0 \in Z_N$ uniformly at random and sets $u = g^a, h = g^b, v = g^c, w = g^d, u_0 = g^{a_0}, h_0 = g^{b_0}, v_0 = g^{c_0}, w_0 = g^{d_0}$. $B$ initially obtains the group elements

$$g, u, h, v, w, X_1 X_2, w^v (Y_2 Y_3)^{w_1}, g^v (Y_2 Y_3)^{w_1}, v^v (Y_2 Y_3)^{w_1}$$

$$u_0, h_0, v_0, w_0, w^{v_0} (Y_2 Y_3)^{w_2}, g^{v_0} (Y_2 Y_3)^{w_2}, v^{v_0} (Y_2 Y_3)^{w_2}$$

(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 $g'$ implicitly set to be $X_1$.

When $A$ requests the challenge ciphertext for messages $M_0, M_1$, identity vector $(I_1' \cdots I_l')$ and $T'$, $B$ makes a ciphertext-type query to the oracle for each $I'_i$ and $T'$. When $B$ makes a ciphertext-type query for some identity $I'_i$, $O$ responds by choosing a random $i'_i \in Z_N$ and returning $((X_1 X_2)^{\delta_i} (X_1 X_2)^{\delta_i'}, (X_1 X_2)^{\psi_1(aT' + b)} (X_1 X_2)^{\psi_1'})$ to $B$. When $B$ makes a ciphertext-type query for some time $T'$, $O$ responds by choosing a random $i'_j \in Z_N$ and returning $((X_1 X_2)^{\psi_1} (X_1 X_2)^{\psi_1'}, (X_1 X_2)^{\psi_1(aT' + b)} (X_1 X_2)^{\psi_1'})$ to $B$. This sets $X^{d_0} = g^{\delta_1}$ and $X^{d_0} = g^{\delta_1}$, which is uniformly random because the value of $d$ and $d_0$ modulo $p_2$ will not appear elsewhere. It implicitly sets $g' = X^{d_0}$. This is identically distributed to a response from $O_6$ and $O_{7/2^2}$, with $a', b'$ equal to $a, b$ modulo $p_2$, and $\sigma_1 = c \bmod p_2, \sigma_2 = c_0 \bmod p_2$. 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 $B$ creates the ESF-1 ciphertexts successfully.

When $B$ requests the secret key of an identity vector $ID = (I_1, \cdots, I_l)$, $B$ creates the ESF-4-SK key by the HIBE-type query responses from $O$ who randomly chooses $y_1, \cdots, y_l, \lambda_1, \cdots, \lambda_{l-1}, r_1, \cdots, r_p, z' \in Z_N$ and generates a ESF-4-SK PSK$_{HIBE,h}$ for every $\theta$

$$K_{i,0} = g^{\psi_i w^i}, K_{i,1} = g^{h_i}, K_{i,2} = g^{o_i}, K_{i,3} = (u^i h)^{\psi_i} v^i, i \in [1, \cdots, j-1]$$

$$K_{j,0} = g^{\psi_j w^j} \sum_{i=1}^{j-1} g^{\psi_i w^i} w^{(Y_2 Y_3)^{\psi_i}}, K_{j,3} = g^{h_j} (Y_2 Y_3)^{\psi_j},$$

$$K_{j,3} = v^i (u^i h)^{\psi_j} (Y_2 Y_3)^{\psi_j}, K_{j,3} = g^{h_j} (Y_2 Y_3)^{\psi_j}$$

When $B$ creates the IBE private key with the index pair $(h, i_j)$ for some time $T$ for the identity vector $(I_1, \cdots, I_{l-1})$ in the index $h$ node, the update key with an index pair $(h, i_j)$ is generated as follows:
1. $i < h$: It randomly chooses $y_0, \ldots, y_{j-1}, \lambda_1, \cdots, \lambda_{j-1}, r'_0, r_1, \cdots, r_{j-1}, z, z' \in \mathbb{Z}_n$ and generates a semi-functional update key $TUK_{ID_b,T,h}$. 

$$U_{0,0} = g^{x_0} \cdot \sum_{i=1}^{j-1} \lambda_i^2 (y_2 Y_3)^{\theta_i} h_i^2, U_{0,1} = g^{y_0} (Y_2 Y_3)^{\nu_0}, U_{0,2} = g^{y_0},$$

$$U_{0,3} = v_0^h (Y_2 Y_3)^{\nu_0} (u_0^h h_0)^{\nu_0}$$

$$U_{0,6} = g^{3h_i^2}, U_{1,1} = g^{y_0}, U_{1,2} = g^{y_0}, U_{1,3} = (u^h)^j w^i, i \in \{1, \cdots, j-1\}$$

2. $i = h$: $B$ chooses random values $y_1, \cdots, y_{j-1}, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_{j-1} \in \mathbb{Z}_n$. $B$ forms the challenge key as:

$$U_{0,0} = g^{x_0} \cdot \sum_{i=1}^{j-1} \lambda_i^2 (y_2 Y_3)^{\theta_i} h_i^2, U_{0,1} = T_1, U_{0,2} = T_3, U_{0,3} = T_3$$

$$U_{0,6} = g^{3h_i^2}, U_{1,1} = g^{y_0}, U_{1,2} = g^{y_0}, U_{1,3} = (u^h)^j w^i, i \in \{1, \cdots, j-1\}$$

where $(T_0, T_1, T_2, T_3)$ is the challenge IBE key queried to $O$ who chooses a random $y_0 \in \mathbb{Z}_N$ and returns $(T_0, T_1, T_2, T_3) = (w_0^h (Y_2 Y_3)^{\nu_0}, g^{y_0} (Y_2 Y_3)^{\nu_0}, v_0^h (Y_2 Y_3)^{\nu_0} T_{h_{\frac{1}{2}}} h, T)$ to $B$.

3. $i > h$: It generates a ESF-4UK as

$$U_{0,0} = g^{x_0} \cdot \sum_{i=1}^{j-1} \lambda_i^2 (y_2 Y_3)^{\theta_i} h_i^2, U_{0,1} = g^{y_0} (Y_2 Y_3)^{\nu_0}, U_{0,2} = g^{y_0} (Y_2 Y_3)^{\nu_0},$$

$$U_{0,3} = v_0^h (u_0^h h_0)^{\nu_0} (Y_2 Y_3)^{\nu_0}$$

$$U_{0,6} = g^{3h_i^2}, U_{1,1} = g^{y_0}, U_{1,2} = g^{y_0}, U_{1,3} = (u^h)^j w^i, i \in \{1, \cdots, j-1\}$$

where $z, z' \in \mathbb{Z}_p$ are randomly chosen.

In the challenge IBE key, it implicitly sets $g'$ 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 $T \in G_{p_2,p_3}$, then this matches the distribution of $O_a$, and so this will be a properly distributed normal key and $B$ is playing Game $F6_{b_{\frac{1}{2}}, \frac{1}{2}}$. If $T \in G$, then this matches the distribution of $O_{b_{\frac{1}{2}}}$ (note random $G_{p_3}$ terms attached to the last two group elements) and then $B$ is playing Game $F6_{b_{\frac{1}{2}}}$.

Hence, if a PPT attacker can distinguish between $F6_{b_{\frac{1}{2}}, \frac{1}{2}}$ and $F6_{b_{\frac{1}{2}}}$ with non-negligible advantage, $O$ can distinguish between $O_b$ and $O_{b_{\frac{1}{2}}}$ with non-negligible advantage. It means $O$ can gain a non-negligible advantage against Assumption 4.

Thus, Under Assumptions 4, no PPT attacker can distinguish between $O_b$ and $O_{b_{\frac{1}{2}}}$ with non-negligible advantage. Thus, no PPT attacker can distinguish between $F6_{b_{\frac{1}{2}}, \frac{1}{2}}$ and $F6_{b_{\frac{1}{2}}}$ with non-negligible advantage.

**Lemma 15** Under Assumptions 3, no PPT attacker can distinguish between $O_{b_{\frac{1}{2}}}$ and $\bar{O}_{b}$ with non-negligible advantage. So no PPT attacker can distinguish between $F6_{b_{\frac{1}{2}}, \frac{1}{2}}$ and $F6_{b_{\frac{1}{2}}}$ with non-negligible advantage.

**Proof** We assume $B$ interacts with one of $O_{b_{\frac{1}{2}}}$, and $\bar{O}_{b}$. $O$ receives $g, g_2, X_1, X_3, T$. $O$ will simulate either $O_{b_{\frac{1}{2}}}$ or $\bar{O}_{b}$ with $B$, depending on the value of $T$ (which is either in $G_{p_3}$ or $G_{p_2,p_3}$). $O$ picks values $a, b, c, d, a_0, b_0, c_0, d_0 \in \mathbb{Z}_N$ uniformly at random and sets $u = g^a$, $h = g^b$, $X_1 = g^{a_0}$, $X_2 = g^{b_0}$, $X_3 = g^{c_0}$, $X_4 = g^{d_0}$, $X_5 = g^{a_0+b_0+c_0+d_0}$.
\( v = g^r, w = g^s, u_0 = g^{s_0}, h_0 = g^{h_0}, v_0 = g^{s_0}, w_0 = g^{h_0}. \) B initially obtains the group elements

\[
\begin{align*}
g, u, h, v, w, & (X_i, X_j)_{g^a}, (X_i, X_j)_{g^b}, (X_i, X_j)_{g^c}, \\
u_0, h_0, v_0, w_0, & (X_i, X_j)_{g^a}, (X_i, X_j)_{g^b}, (X_i, X_j)_{g^c}
\end{align*}
\]

\( (0) \)

from its oracle simulator, and gives A the public parameters in Eq 83.

When A requests the challenge ciphertext for messages \( M_0, M_1, \) identity vector \( (I_1', \cdots, I_j') \) and \( T', B \) makes a ciphertext-type query to the oracle for each \( I_i' \) and \( T' \). When B makes a ciphertext-type query for some identity \( I \), \( O \) responds by choosing a random \( t + Z_N \) and returning \( (w, y, z), (u, h, g, (u' h) )_{g^a} \) to \( B \) as same as Eq 10. When \( B \) makes a ciphertext-type query for some time \( T \), \( O \) responds by choosing a random \( t + Z_N \) and returning \( (w, y, z), (u, h, g, (u' h) )_{g^a} \) to \( B \). Then \( B \) creates the ESF-1 ciphertexts successfully.

When A requests the secret key of an identity vector \( I = (I_1, \cdots, I_j) \), B creates the ESF-4-SK by the HIBE-type query response from \( O \) and the secret key for \( I \) in some node \( \theta \) is

\[
K_{ib} = g^{r_i \nu_i}, K_{i,1} = g^{h_i}, K_{i,2} = g^{r_i}, K_{i,3} = (u' h)^{r_i \nu_i}, i \in \{1, \cdots, j - 1\}
\]

\[
K_{j,0} = g^{r_j \nu_j} \sum_{i=1}^{j-1} (X_i X_j g_2)^{y_{i,j}} (X_i X_j g_2)^{y_i}, K_{j,1} = (X_i X_j g_2)^{y_i},
\]

\[
K_{j,2} = (X_i X_j g_2)^{y_{j,0}} (X_i X_j g_2)^{y_j}, K_{j,3} = (X_i X_j g_2)^{y_j} (X_i X_j g_2)^{y_j h_i g_2}
\]

where \( \lambda_j, \lambda_{j-1}, \lambda_{j-1}, \cdots, \lambda_2, r_1, \cdots, r_{j-1}, r', z, z' \in Z_N \) are randomly chosen.

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

1. \( i < h \): 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 \( TK_{I, T, \theta h} \)

\[
U_{0,0} = g^{r_0 \nu_0} \sum_{i=1}^{j-1} (X_i X_j g_2)^{y_{i,j}} (X_i X_j g_2)^{y_i}, U_{0,1} = (X_i X_j g_2)^{y_i}, U_{0,2} = g^{s_0},
\]

\[
U_{0,3} = (X_i X_j g_2)^{y_i} (u'_h h_0)^{y_i}
\]

\[
U_{0,0} = (X_i X_j g_2)^{y_i} (u'_h h_0)^{y_i}
\]

\[
U_{i} = g^{r_i \nu_i}, U_{i,1} = g^{h_i}, U_{i,2} = g^{r_i}, U_{i,3} = (u' h)^{r_i \nu_i}, i \in \{1, \cdots, j - 1\}
\]

2. \( i = h \): B chooses random values \( y_1, \cdots, y_{j-1}, \lambda_1, \cdots, \lambda_{j-1}, r_1, \cdots, r_{j-1} \in Z_N \). B forms the challenge key as:

\[
U_{0,0} = g^{r_0 \nu_0} \sum_{i=1}^{j-1} (X_i X_j g_2)^{y_{i,j}} (X_i X_j g_2)^{y_i}, U_{0,1} = T_0, U_{0,2} = T_3, U_{0,3} = T_2
\]

\[
U_{i} = g^{r_i \nu_i}, U_{i,1} = g^{h_i}, U_{i,2} = g^{r_i}, U_{i,3} = (u' h)^{r_i \nu_i}, i \in \{1, \cdots, j - 1\}
\]

where \( (T_0, T_1, T_2, T_3) \) is the challenge IBE key queried to \( O \) who chooses a random \( y_0 \in Z_N \) and returns \( (T_0, T_1, T_2, T_3) = ((X_i X_j g_2)^{y_{i,j}}, (X_i X_j g_2)^{y_i}, (X_i X_j g_2)^{y_i} T_0^{v_i h_i g_2}) \). This implicitly sets \( g' \) to be the \( G_F \) part of \( T \).
3. \(i_c > h_c\): It generates a ESF-4-USK as

\[
U_{0,0} = g^{s_{y_0} - \sum_{i=1}^{j} h_i (X, X_d g_2)^{s_{y_0}}}, U_{0,1} = (X, X_d g_2)^{y_0},
\]

\[
U_{0,2} = (X, X_d)_{y_0} g_2^{z'}, U_{0,3} = (X, X_d g_2)^{y_0} (X, X_d)_{y_0 h_{k=1} h_{k=1}} g_2^{z'}
\]

\[
U_{i,0} = g^{u_i w_{0,i}}, U_{i,1} = g^{u_i}, U_{i,2} = (u_i h)^{y_0}, U_{i,3} = (u_i h)^{y_0}, i \in \{1, \cdots, j - 1\}
\]

where \(z, z' \in Z_p\) are randomly chosen.

In the challenge IBE key, it implicitly sets \(g^{y_0}\) to be the \(G_p\) part of \(T\). We note that \(a_0, b_0\) modulo \(p_2, p_3\) are uniformly random and do not appear elsewhere. If \(T \in G_{p,p3}\) then this matches the distribution of \(O_{7/2}\), and so this will be a properly distributed normal key and \(B\) is playing Game \(F_{6_{h,1}}\). If \(T \in G_{p,3}\) then this matches the distribution of \(I_{\tilde{h}}\) and then \(\mathcal{B}\) is playing Game \(F_{6_{h,2}}\).

Hence, if a PPT attacker can distinguish between \(F_{6_{h,1}}\) and \(F_{6_{h,2}}\) with non-negligible advantage, \(\mathcal{O}\) can distinguish between \(O_{7/2}\) and \(\tilde{O}_{\tilde{h}}\) 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 \(\tilde{O}_{\tilde{h}}\) with non-negligible advantage. Thus, no PPT attacker can distinguish between \(F_{6_{h,1}}\) and \(F_{6_{h,2}}\) with non-negligible advantage.

**Lemma 16** Under Assumptions 3, no PPT attacker can distinguish between \(\tilde{O}_{\tilde{h}}\) and \(\tilde{O}_{\tilde{h}}\) with non-negligible advantage. So no PPT attacker can distinguish between \(S_{\tilde{h},1}\) and \(S_{\tilde{h},2}\) with non-negligible advantage.

**Proof** We assume \(B\) interacts with one of \(\tilde{O}_{\tilde{h}}, \tilde{O}_{\tilde{h}}\). \(O\) receives \(g, g_2, X, X_d, T\). \(O\) will simulate either \(\tilde{O}_{\tilde{h}}\) or \(\tilde{O}_{\tilde{h}}\) with \(B\), depending on the value of \(T\) (which is either in \(G_{p_1}\) or \(G_{p_3}\)). \(B\) initially obtains the group elements in Eq 82

\[
g, u, h, v, w, g^{y_0}, (X, X_d)_{y_0} g_2^{z'}, (X, X_d)_{y_0} g_2^{z'}, (X, X_d)_{y_0} g_2^{z'}
\]

\[
u_0, h_0, v_0, w_0, (X, X_d)_{y_0} g_2^{z'}, (X, X_d)_{y_0} g_2^{z'}, (X, X_d)_{y_0} g_2^{z'}
\]

from its oracle simulator.

When \(A\) requests the secret key of an identity vector \(ID_j = (I_1, \cdots, I_j)\), \(B\) creates the ESF-4-USK key by the HIBE-type query response from \(O\) and the secret key for \(ID_j\) in some node \(\theta\) is

\[
K_{j,0} = g^{u_{j,0} w_{j,0}}, K_{j,1} = g^{u_{j,0}}, K_{j,2} = g^{u_{j,0}}, K_{j,3} = (u_{j,0} h)^{y_0}, i \in \{1, \cdots, j - 1\}
\]

\[
K_{j,0} = g^{u_{j,0} w_{j,0} h_{j,0}}, K_{j,1} = (X, X_d)_{y_0} g_2^{z'}, K_{j,2} = (X, X_d)_{y_0} g_2^{z'}, K_{j,3} = (X, X_d)_{y_0} g_2^{z'}
\]

where \(y_j, \cdots, y_{j-1}, y_{j-1}, \cdots, y_{j-1}, r_{j-1}, r_{j-1}, r_{j-1}, r_{j-1}, r'_{j-1}, z, z' \in Z_p\) are randomly chosen.

When \(A\) requests the update key of an identity vector \(ID_j = (l_1, \cdots, l_j)\) and the same \(T\), \(B\) creates the UK key by the IBE-type query response from \(O\) and the secret key for \(ID_j\) in some node \(\theta\) is generated as follows: \(O\) randomly chooses \(y_{j,1}, \cdots, y_{j-1}, \lambda_{j-1}, \lambda_{j-1}, \cdots, \lambda_{j-1}, r_0, r_1, \cdots, r_{j-1}\),
If $z, z' \in \mathbb{Z}_n$ and generates a semi-functional update key $TK_{ID_0,T,B_0}$

$$U_{0,0} = g^{a^{-1}h - \sum_{i=1}^{i-1} h(X_iX_gz_i^2h)}/, U_{0,1} = (X_1X_2g)\gamma_0^h, U_{0,2} = g^{\gamma_0},$$

$$U_{0,3} = (X_1X_2g)\tau_0^h (u_0^h h_0)^h$$

$$U_{i,0} = g^{g_iw_i}, U_{i,1} = g^{\gamma_i}, U_{i,2} = (u_i^h h)^\gamma v_i^i, i \in \{1, \cdots, i - 1\}$$

When $A$ requests the challenge ciphertext for messages $M_0, M_1$, identity vector $(I_{i1}, \cdots, I_{ik})$ and $T, B$ makes a ciphertext-type query to the oracle for each $I_{i1}$ and $T$. In response to each query for $I_{i1}$ or $T$, $O$ gets random $t_0, t_1, \ldots, t_i \in \mathbb{Z}_p$ chooses $\beta \in \{0, 1\}$ and creates the ciphertext as

$$(C = M_p \epsilon((g^x_{g_2}, g)^\gamma, C_0 = g^{\gamma_2}, \{C_{i,1}, C_{i,2}, C_{i,3}\}^{i \in 0})$$

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

1. $i < k$:
   If $i = 0$, $O$ responds with the ciphertext-element-group
   $$(w_i^\gamma g_2\tau_{2i}^h, (u_i^h h)_2^{(d_i + t_i)^h}, g^{(d_i + t_i)^h} g_2^{(d_i + t_i)^h})$$
   else the element group is
   $$(w_i^\gamma g_2^v, g_2^{(d_i + t_i)^h}, g^{(d_i + t_i)^h})$$
   This is identically distributed to a response from $O_2$.

2. $i = k$:
   $O$ chooses $z \in \mathbb{Z}_p$ randomly and responds with the ciphertext-element-group $(T_1, T_2) = (w_i^\gamma g_2^t, T_3, T_4, T_5)$ if $k > 0$.
   Else if $k = 0$, $O$ responds with
   $$(w_i^\gamma g_2^t, g_2^{(d_i + t_i)^h}, T_3, T_4, T_5)$$
   We note that the $G_{p_i}$ parts here are properly distributed, since $a_1 = c$ modulo $p_2$ and $a_2 = c_0$ modulo $p_2$.

3. $i > k$:
   The ciphertext-element-group is $(w_i^\gamma g_2^t, (u_i^h h)_2^{(d_i + t_i)^h}, g^{(d_i + t_i)^h})$.
   This is identically distributed to a response from $O_1$.

When $T \in G_{p_i}$ the values of $a, b$ modulo $p_3$ only appear in the response to the challenge key-type query, which means that the $G_{p_i}$ 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, $O$ has properly simulated the responses of $O^{k}_1$ and this will be a properly distributed EST-2$k$-CT and so $B$ is playing Game $S_{k,1}$.

When $T \in G_{p_p, p_3}$ we must argue that the values $al + b$ and $al_i + b$ appear uniformly random modulo $p_3$; this follows by pairwise independence of $al + b$ as a function of $I$ modulo $p_3$, since we have restricted the Type-1 adversary to choose $I$ and $I^+_i$ so that $I \neq I^+_i$ modulo $p_3$ and $a, b$ modulo $p_3$ only appear in these two values. Hence, $O$ has produced a properly distributed EST-4$k$-CT and $O$ has properly simulated the response of $O^{k}_i$ in this case. So $B$ is playing Game $S_{k,3}$. We have thus shown that $O$ can use the output of $B$ 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, $O$ can distinguish the corresponding pair between $\tilde{O}^{k}_i$ and $\tilde{O}^{k}_i$ with non-negligible advantage. It means $O$ can use the output of $B$ to achieve a non-negligible advantage against Assumption 3.

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

Proof We assume $B$ interacts with one of $\tilde{O}_k^*$, $\tilde{O}_k^*$. $O$ receives $g, g_1, X_1, X_2, Y_2, Y_3, T$. $O$ will simulate either $\tilde{O}_k^*$ or $\tilde{O}_k^*$ with $B$, depending on the value of $(T)$ (which is either in $G$ or $G_{\beta_1, \beta_2}$). $O$ picks values $a, b, c, d, a_0, b_0, c_0, d_0 \in Z_\lambda$ uniformly at random and sets $u = g^u, h = g^h, v = g^v, w = g^w, u_0 = g^{u_0}, h_0 = g^{h_0}, v_0 = g^{v_0}, w_0 = g^{w_0}$. $B$ initially obtains the group elements in Eq 85

$$g, u, v, w, X, X_2, w'(Y_2, Y_3), g'(Y_2, Y_3), v'(Y_2, Y_3)$$

$$u_0, v_0, w_0, w_0'(Y_2, Y_3), g_0'(Y_2, Y_3), v_0'(Y_2, Y_3)$$

from its oracle simulator where $z, y_0, y, y_0 \in Z_p$ are randomly chosen.

When $A$ makes a secret key query for the identity ID$_{I_0} = (I_1, \ldots, I_3)$, then $B$ makes its challenge HIBE-key-type query for $I_0$. $O$ chooses $r, y'_{0}, z^{'}, z^{'}, m_0, m_0 \in Z_\nu$ randomly and returns the group elements

$$(w'(Y_2, Y_3), g'(Y_2, Y_3), v'(Y_2, Y_3))$$

to $B$. And then $B$ creates the ESF-4 secret key by using the group elements.

Upon receiving a challenge IBE-key-type query for $T \in Z_\nu, O$ chooses $r_0, y_0 \in Z_\nu$ randomly and returns the group elements

$$(w_0'(Y_2, Y_3), g_0'(Y_2, Y_3), v_0'(Y_2, Y_3))$$

to $B$. And then $B$ creates the semi-functional update key by using the group elements.

When $A$ requests the challenge ciphertext for messages $M_0, M_1$, identity vector $(I_1, I_2, I_3)$ and $T$, $B$ makes a ciphertext-type query to the oracle for each $I_0$. In response to each query for $I_0$ or $T$, $O$ gets random $r_0, r_0', \ldots, r_0', t_0, t_0, t_0', \ldots, t_0, b_0 \in \{0, 1\}$ and creates the ciphertext as

$$(C = M \beta \epsilon(X, X_2, g)^v, C_0 = X_1, X_2, \{C_{1,1}, C_{1,2}, C_{1,3}\}_{j=1}^3)$$

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

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

$$(X_1 X_2)^{d_0}(X_2 X_3)^{a_0 + b_0}, (X_1 X_2)^{a_0}(a_0 + b_0), (X_1 X_2)^{a_0})$$

else the ciphertext-element-group is

$$(X_1 X_2)^{d_0}(X_1 X_2)^{a_0}, (X_1 X_3)^{a_0 + b_0}, (X_1 X_2)^{a_0})$$

This sets $X_2^3 = g^3$ and $X_2^{d_0} = g^{d_0}$, which is uniformly random because the value of $d$ and $d_0$ modulo $p_2$ will not appear elsewhere. It implicitly sets $g^c = X_1^c$. This is identically distributed to a response from $O_2$, with $a'$, $b'$ equal to $a, b$ modulo $p_2$, and $a_1 = c$ modulo $p_2$, $a_2 = c_0$ 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 $(T_1, T_3, T_2) = ((X_1 X_2)^{d_0} T_0, T_0^d, T^d + b_0), (T)$; If $k > 0$, the ciphertext-element-group is $(T_1, T_3, T_2) = ((X_1 X_2)^{d_0} T^c, T^c, T^c + b_0), (T)$;
3. \( i > k \): The ciphertext-element-group is \((X, X_j)^i H, (u^i H)^i, g^i)\).

If \( T \in G_{p, p^3} \), then the response for ciphertext-type query \( i \) is identically distributed to a response from \( O^i \).

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

If \( T \in G \), then this response additionally has terms in \( G_p \), which are appropriately distributed with \( c = \sigma_a, a = a', b = b' \) modulo \( p_2 \) or \( c_0 = \sigma_a a_0 = a', b_0 = b' \) modulo \( p_2 \). Thus, the response is identically distributed to a response from \( O^i \). In this case, \( O \) has produced a properly distributed EST-4\( k \)-CT and \( B \) 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, \( O \) can distinguish the corresponding pair between \( O^i \) and \( O^i \) with non-negligible advantage. It means \( O \) can use the output of \( B \) to achieve a non-negligible advantage against Assumption 4.

Thus, Under Assumptions 4, no PPT attacker can distinguish between \( O^i \) and \( O^i \) 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 \( O^i \) and \( O^i \) 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 \( B \) interacts with one of \( O^i \), \( O^i \). \( O \) receives \( g, g_2, X_1 X_3, Y, Y_3, T \). \( O \) will simulate either \( S_{k, 1} \) or \( S_{k, 1} \) with the value of \( T \) (which is either in \( G_p \) or \( G_{p, p^3} \), \( B \) initially obtains the group elements in Eq 82

\[
g, u, h, v, w, g^e g_2^e, (X_1 X_3)^i g_2^{i e}, (X_1 X_3)^i g_2^{j e}, (X_1 X_3)^i g_2^{j e}
\]

\[
u_0, v_0, w_0, (X_1 X_3)^i g_2^{i e}, (X_1 X_3)^j g_2^{j e}, (X_1 X_3)^i g_2^{i e}
\]

from its oracle simulator. It chooses \( \alpha \in Z_n \) randomly, and gives \( A \) the public parameters in Eq 83. \( B \) can respond by using the normal update key generation and the normal decryption key derivation algorithm, since it knows \( \alpha \).

When \( A \) requests the secret key of an identity vector \( ID_0 = (I_1, \cdots, I_j) \), \( B \) creates the ESF-4\( k \)-key by the HIBE-type query response from \( O \) and the secret key for \( ID_0 \) in some node \( \theta \) is

\[
K_{i, 0} = g^{i e}, K_{i, 1} = g^{i e}, K_{i, 2} = g^{i e}, K_{i, 3} = (u^i h)^i g^{j e}, i \in \{1, \cdots, j - 1\}
\]

\[
K_{i, 0} = g^{i e} \sum_{m+1}^{m+1}(X_1 X_3 g_2)^{d_i}, K_{i, 1} = (X_1 X_3)^{d_i}, K_{i, 2} = (X_1 X_3)^{d_i} g_2^{d_i}, K_{i, 3} = (X_1 X_3)^{d_i} g_2^{d_i}
\]

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

When \( A \) requests the update key of an identity vector \( ID_0 = (I_1, \cdots, I_j) \) and the imn \( T \), \( B \) creates the UK key by the IBE-type query response from \( O \) and the secret key for \( ID_0 \) in some node \( \theta \) is generated as follows: \( O \) randomly chooses \( y'_i, y'_i, \cdots, y'_{j-1}, \lambda_1, \cdots, \lambda_{j-1}, r_0, r_1, \cdots, r_{j-1} \),
z, z' \in \mathbb{Z}_n \text{ and generates a semi-functional update key } \text{TUK}_{ID_k,T,B_h}

\begin{align*}
U_{0,0} &= g^{\nu_z} \cdot \prod_{i=1}^{j-1} (X_i X_i g_2) \cdot \nu_{z_i}, \quad U_{0,1} = (X_i X_i g_2)^{\nu_z}, \quad U_{0,2} = g^{\nu_z}, \\
U_{0,3} &= (X_i X_i g_2)^{\nu_z} (u_i h_i)^{\nu_z} \\
U_{i,0} &= g^{\nu_z} w^i, \quad U_{i,1} = g^h, \quad U_{i,2} = g^{\nu_z}, \quad U_{i,3} = (u^i h^i)^{\nu_z}, \quad i \in \{1, \ldots, j-1\}
\end{align*}

When A requests the challenge ciphertext for messages M_0, M_1, identity vector (I_1, \ldots, I_j) and T, B makes a ciphertext-type query to the oracle for each I_i \neq T. In response to each query for I_i or T, O gets random t_0, t_1, \ldots, t_i \in \mathbb{Z}_p \text{ chooses } \beta \in \{0, 1\} \text{ and creates the ciphertext as}

\[(C = M_\beta e(g^{\nu_z}, g)^z, C_0 = g^{\nu_z}, \{C_{i,1}, C_{i,2}, C_{i,3}\})_{i=0}^j\]

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

1. \(i < k\): If \(i = 0\), O responds with the ciphertext-element-group \((w_i g_1^{\nu_i} v_i g_2^{\nu_i}, (u_i^T h_i)^{\nu_i} g_2^{\nu_i}, g^{\nu_i})\), else the element group \((w_i g_1^{\nu_i} v_i g_2^{\nu_i}, (u_i^T h_i)^{\nu_i} g_2^{\nu_i}, g^{\nu_i})\);

2. \(i = k\): The ciphertext-element-group is \((T_1, T_3, T_2) = (w_i g_1^{\nu_i} T^e, T^{u_i^T} h_i, T^{v_i})\);

3. \(i > k\): The ciphertext-element-group is \((w_i g_1^{\nu_i} v_i, (u_i^T h_i)^{\nu_i}, g^h)\).

We must now argue that the challenge key-type query and the \(k\)th ciphertext-type query responses are properly distributed. If \(T \in G_{p_3}\), then the response to the \(k\)th 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 \(G_{p_2}\). This will be a properly distributed EST-\(2^k\)-CT which means that the responses of \(O\) properly simulate the responses of \(O_{k-1}\) and \(B\) is playing Game \('S_{k-1,1}'\).

If \(T \not\in G_{p_3}\), then we must argue that \(al + b\) and \(al_i + b\) both appear to be uniformly random modulo \(p_3\); this follows from pairwise independence of the function \(al + b\) modulo \(p_3\), since we have restricted the Type-1 adversary to choose \(I\) and \(I_i\) so that \(I \neq I_i\) 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, \(O\) has produced a properly distributed EST-\(3^k\)-CT which means that \(O\) has properly simulated the responses of \(O_{k-1}\) and \(B\) is playing Game \('S_{k,3}'\).

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

Thus, Under Assumptions 3, no PPT attacker can distinguish between \(O_{k-1}\) and \(O_{k,1}\) 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 \(O_{k,1}\) and \(O_{k,2}\) with non-negligible advantage. So no PPT attacker can distinguish between \(I_{h-1,2}\) and \(I_{h,1}\) with non-negligible advantage.
We assume $B$ interacts with one of $O^*_0$, $O^*_{1/2}$. $O$ receives $g$, $g^h$, $X_1$, $X_2$, $Y_1$, $Y_2$, $T$. $O$ will simulate either $O^*_0$ or $O^*_{1/2}$ with $B$, depending on the value of $T$ (which is either in $G$ or $G_{p,2}$). $O$ picks values $a$, $b$, $c$, $d$, $a_0$, $b_0$, $c_0$, $d_0 \in Z_n$ uniformly at random and sets $u = g^a$, $h = g^b$, $v = g^c$, $w = g^d$, $u_0 = g^{a_0}$, $h_0 = g^{b_0}$, $v_0 = g^{c_0}$, $w_0 = g^{d_0}$. $B$ initially obtains the group elements in Eq 89

$$g, u, h, v, w, X_1, X_2, w'(Y_2 Y_3)^{\gamma_1}, g'(Y_2 Y_3)^{\gamma_2}, v'(Y_2 Y_3)^{\gamma_3}$$

$$u_0, h_0, v_0, w_0, v_0'(Y_2 Y_3)^{\gamma_2}, g'(Y_2 Y_3)^{\gamma_2}, v_0'(Y_2 Y_3)^{\gamma_2}$$

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 $g'$ implicitly set to be $X_1$.

When $A$ requests the challenge ciphertext for messages $M_0$, $M_1$, identity vector $(I_1^*, \cdots, I^n^*)$ and $T^*$, $B$ makes a ciphertext-type query to the oracle for each $I^*_i$ and $T^*$. When $B$ makes a ciphertext-type query for some identity $I^*_i$, $O$ responds by choosing a random $t'_i \in Z_n$ and returning $((X_i X'_i)^{\gamma_3} v_i, (\psi_i^* h_i)^{\gamma_3}, g_i^\gamma_i)$ to $B$. When $B$ makes a ciphertext-type query for some time $T^*$, $O$ responds by choosing a random $r_0 \in Z_n$ and returning $((X_1 X_2)^{\gamma_0} v_0, (\psi_0^* h_0)^{\gamma_0}, g_0^\gamma_0)$ to $B$. (Note that this implicitly sets $g_0^\gamma_0 = X_2^\gamma_0$ and $g_2^\gamma_2 = X_2^\gamma_2$, which is uniformly random because the value of $d$ and $d_0$ modulo $p_2$ does not occur elsewhere.) Then $B$ creates the semi-functional ciphertexts successfully.

Upon receiving a challenge IBE-key-type query for $T \in Z_n$, $O$ chooses $r_0, y_0 \in Z_n$ randomly and returns the group elements

$$(w_0^\gamma (Y_2 Y_3)^{\gamma_0}, g_0^\gamma (Y_2 Y_3)^{\gamma_0}, g_0^\gamma, v_0^\gamma (Y_2 Y_3)^{\gamma_0} (u_0^\gamma h_0)^{\gamma_0})$$

of $B$. And then $B$ creates the semi-functional update key by using the group elements.

When $B$ creates the HIBE private key with the index pair $(h, i)$ for the identity vector $(I_1, \cdots, I_{j-1})$ in the index $h$ node, the secret key with an index pair $(h, i)$ is generated as follows:

1. $i < h$: 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 $TUK^b_{D_{h,i}, T,h'}$

$$K_{ij} = g_i^h w_i^h, K_{i1} = g_i^h, K_{i2} = g_i^h, K_{i3} = (u_i h_i)^{\gamma_3} w_i^h, i \in \{1, \cdots, j - 1\}$$

$$K_{j0} = g_i^{\gamma_0} \cdot \sum_{i=1}^{j-1} w_i^{\gamma_0} (Y_2 Y_3)^{\gamma_0}, K_{j1} = g_i^h (Y_2 Y_3)^{\gamma_1}, K_{j2} = g_i^{\gamma_2}, K_{j3} = v_i^{\gamma_3} (Y_2 Y_3)^{\gamma_3} (u_i h_i)^{\gamma_3}$$

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

$$K_{0} = g_i^{h} w_i^{h}, K_{1} = g_i^{h}, K_{2} = g_i^{h}, K_{3} = (u_i h_i)^{\gamma_3} w_i^{h}, i \in \{1, \cdots, j - 1\}$$

$$K_{0} = g_i^{\gamma_0} \cdot \sum_{i=1}^{j-1} T_0, K_{1} = T_1, K_{2} = T_2, K_{3} = T_2$$

where $(T_0, T_1, T_2, T_3)$ is the challenge HIBE key queried to $O$ who chooses a random $y_j \in Z_N$ and returns $(T_0, T_1, T_2, T_3) = (w_0^{\gamma_0} (Y_2 Y_3)^{\gamma_0}, g_0^{\gamma_0} (Y_2 Y_3)^{\gamma_0}, v_0^{\gamma_3} (Y_2 Y_3)^{\gamma_3} T_0^{\gamma_0}, T)$ to $B$. 

3. \( i_e > h: \) It generates a ESF-4-SK as
\[
K_{g0} = g^{h \cdot w^0}, K_{i1} = g^{v_i}, K_{i2} = g^{v_0}, K_{i3} = (u^i h)^v, i \in \{1, \cdots, j - 1\}
\]
\[
K_{g0} = g^{g^0 - \sum_{i=1}^{j-1} (w^i (Y_1 Y_j)^{g^0}), K_{i1} = g^{g^0} (Y_2 Y_3)^{g^0}, K_{i2} = g^{g^0} (Y_2 Y_4)^{g^0},
\]
\[
K_{i3} = v^i (u^i h)^v (Y_2 Y_4)^{g^0}
\]

where \( z, z' \in Z_p \) are randomly chosen.

In the challenge HIBE key, it implicitly sets \( g_i \) to be the \( G_p \) part of \( T \). We note that \( a, b \) modulo \( p_2, p_3 \) are uniformly random and do not appear elsewhere. If \( T \in G_{p_2, p_3} \), then this matches the distribution of \( \tilde{O}_0 \), and so this will be a properly distributed ESF-4-SK key and \( B \) is playing Game \( I_{h-2} \). If \( T \in G \), then this matches the distribution of \( O_{7/2} \) (note random \( G_p \) terms attached to the last two group elements) and then \( B \) is playing Game \( I_{h,1} \).

Hence, if a PPT attacker can distinguish between \( I_{h,1} \) and \( I_{h,1} \) with non-negligible advantage, \( O \) can distinguish between \( \tilde{O}_0 \) and \( O_{7/2} \) with non-negligible advantage. It means \( O \) can gain a non-negligible advantage against Assumption 4.

Thus, Under Assumptions 4, no PPT attacker can distinguish between \( \tilde{O}_0 \) and \( O_{7/2} \) with non-negligible advantage. Thus, no PPT attacker can distinguish between \( I_{h-2} \) and \( I_{h,1} \) 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 \( I_{h,1} \) and \( I_{h,2} \) with non-negligible advantage.

**Proof** We assume \( B \) interacts with one of \( O_{7/2} \), and \( O_4 \), \( O \) receives \( g, g_2, x_1, x_3, T \). \( O \) will simulate either \( O_{7/2} \) or \( O_4 \) with \( B \), depending on the value of \( T \) (which is either in \( G_p \) or \( G_{p_1, p_3} \)).

\( O \) picks values \( a, b, c, d, a_0, b_0, c_0, d_0 \in Z_\text{P} \) uniformly at random and sets \( u = g^a, h = g^b, v = g^c, w = g^d, u_0 = g^{a_0}, h_0 = g^{b_0}, v_0 = g^{c_0}, w_0 = g^{d_0} \). \( B \) initially obtains the group elements in Eq 90

\[
g, g, u, h, v, w, g^{g_2}, (x_1 x_3)^{g_2}, (x_1 x_3)^{g_2}, (x_3 x_1)^{g_2}, (x_3 x_1)^{g_2},
\]
\[
u_0, h_0, v_0, w_0, (x_1 x_3)^{g_2}, (x_1 x_3)^{g_2}, (x_3 x_1)^{g_2}, (x_3 x_1)^{g_2},
\]

from its oracle simulator, and gives \( A \) the public parameters in Eq 83.

When \( A \) requests the challenge ciphertext for messages \( M_0, M_1 \), identity vector \((I_1', \cdots, I_{j-1}')\) and \( T' \), \( B \) makes a ciphertext-type query to the oracle for each \( I_i' \) and \( T' \). When \( B \) makes a ciphertext-type query for some identity \( I' \), \( O \) responds by choosing a random \( t \in Z_\text{P} \) and returning \((w^t g_2, v^t, (u^t h)^v)\) to \( B \). When \( B \) makes a ciphertext-type query for some time \( T' \), \( O \) responds by choosing a random \( t_0 \in Z_\text{P} \) and returning \((w^t g_2, v_0, (u^t h_0)^v)\) to \( B \). Then \( B \) creates the semi-functional ciphertexts successfully.

Upon receiving a challenge IBE-key-type query for \( T' \in Z_{m} \), \( O \) chooses \( r_0, y_0' \in Z_n \) randomly and returns the group elements

\[
(x_1 x_2 g_2)^{g_0}, (x_1 x_2 g_2)^{g_0}, (x_1 x_2 g_2)^{g_0}, (u^t h_0)^v
\]

to \( B \). And then \( B \) creates the semi-functional update key by using the group elements.

When \( B \) creates the HIBE private key with the index pair \((h, i)\) for the identity vector \((I_1, \cdots, I_{j-1})\) in the index \( h \) node, the secret key with an index pair \((h, i)\) is generated as follows:
1. \(i_c < h_2\): It randomly chooses \(y'_1, y'_2, \ldots, y'_{j-1}, \lambda_1, \ldots, \lambda_{j-1}, r_1, \ldots, r_{j-1}, z, z' \in \mathbb{Z}_p\) and generates a semi-functional secret key \(PSK_{ID, h_2}\)

\[
K_{b,0} = g^{h_2}w_0, K_{b,1} = g^{y_1}, K_{b,2} = g^{y_2}, K_{b,3} = (u^h)^{y_1}w_0, i \in \{1, \ldots, j-1\}
\]

\[
K_{b,0} = g^{h_2} \sum_{i=1}^{j-1} (X_iX_ig_i)^{\theta_i}, K_{b,1} = (X_iX_ig_i)^{y_i}, K_{b,2} = g^{y_2}, K_{b,3} = (X_iX_ig_i)^{y_i}(u^h)^{y_1}
\]

2. \(i_c = h_2\): \(B\) chooses random values \(y_1, \ldots, y_{j-1}, \lambda_1, \ldots, \lambda_{j-1}, r_1, \ldots, r_{j-1} \in \mathbb{Z}_p\), \(B\) forms the challenge key as:

\[
K_{b,0} = g^{h_2}w_0, K_{b,1} = g^{y_1}, K_{b,2} = g^{y_2}, K_{b,3} = (u^h)^{y_1}w_0, i \in \{1, \ldots, j-1\}
\]

\[
K_{b,0} = g^{h_2} \sum_{i=1}^{j-1} T_i, K_{b,1} = T_1, K_{b,2} = T_2, K_{b,3} = T_3
\]

where \((T_0, T_1, T_2, T_3)\) is the challenge HIBE key queried to \(O\) who chooses a random \(y'_i \in \mathbb{Z}_N\) and returns \((T_0, T_1, T_2, T_3) = ((X_iX_ig_i)^{y_i}, (X_iX_ig_i)^y, (X_iX_ig_i)^{y_i}T_i^{a_{i+b}}, T)\). This implicitly sets \(g^{y_i}\) to be the \(G_p\) part of \(T\).

3. \(i_c > h_2\): It generates a ESF-4-4 SK as

\[
K_{b,0} = g^{h_2}w_0, K_{b,1} = g^{y_1}, K_{b,2} = g^{y_2}, K_{b,3} = (u^h)^{y_1}w_0, i \in \{1, \ldots, j-1\}
\]

\[
K_{b,0} = g^{h_2} \sum_{i=1}^{j-1} (X_iX_ig_i)^{\theta_i}, K_{b,1} = (X_iX_ig_i)^{y_i}, K_{b,2} = (X_iX_ig_i)^{y_i}g_i^z, K_{b,3} = (X_iX_ig_i)^{y_i}(u^h)^{y_1}g_i^z
\]

where \(z, z' \in \mathbb{Z}_p\) are randomly chosen.

In the challenge HIBE key, it implicitly sets \(g^{y_i}\) to be the \(G_p\) part of \(T\). We note that \(a, b\) modulo \(p_2, p_1\) are uniformly random and do not appear elsewhere. If \(T \in G_{p_2, p_1}\), then this matches the distribution of \(O_{\gamma_2}\), and so this will be a properly distributed normal key and \(B\) is playing Game \(I_{h_1}\). If \(T \in G_{p_1}\), then this matches the distribution of \(O_4\) and then \(B\) is playing Game \(I_{h_2}\).

Hence, if a PPT attacker can distinguish between \(I_{h_1}\) and \(I_{h_2}\), with non-negligible advantage, \(O\) can distinguish between \(O_{\gamma_2}\) and \(O_4\) with non-negligible advantage. It means \(O\) can gain a non-negligible advantage against Assumption 3.

Thus, Under Assumptions 3, no PPT attacker can distinguish between \(O_{\gamma_2}\) and \(O_4\) with non-negligible advantage. Thus, no PPT attacker can distinguish between \(I_{h_1}\) and \(I_{h_2}\) 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, Lotspeich 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 of CCA-Secure Revocable 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.