New Revocable IBE in Prime-Order Groups: Adaptively Secure, Decryption Key Exposure Resistant, and with Short Public Parameters*

Yohei Watanabe[1,2,†], Keita Emura[3], and Jae Hong Seo[4]

[1] The University of Electro-Communications, Tokyo, Japan

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

[3] National Institute of Information and Communications Technology (NICT), Tokyo, Japan

[4] Myongji University, Yongin, Korea

watanabe@uec.ac.jp, k-emura@nict.go.jp, jaehongseo@mju.ac.kr

April 1, 2017

Abstract

Revoking corrupted users is a desirable functionality for cryptosystems. Since Boldyreva, Goyal, and Kumar (ACM CCS 2008) proposed a notable result for scalable revocation method in identity-based encryption (IBE), several works have improved either the security or the efficiency of revocable IBE (RIBE). Currently, all existing scalable RIBE schemes that achieve adaptively security against decryption key exposure resistance (DKER) can be categorized into two groups; either with long public parameters or over composite-order bilinear groups. From both practical and theoretical points of views, it would be interesting to construct adaptively secure RIBE scheme with DKER and short public parameters in prime-order bilinear groups.

In this paper, we address this goal by using Seo and Emura's technique (PKC 2013), which transforms the Waters IBE to the corresponding RIBE. First, we identify necessary requirements for the input IBE of their transforming technique. Next, we propose a new IBE scheme having several desirable properties; satisfying all the requirements for the Seo-Emura technique, constant-size public parameters, and using prime-order bilinear groups. Finally, by applying the Seo-Emura technique, we obtain the first adaptively secure RIBE scheme with DKER and constant-size public parameters in prime-order bilinear groups.

Keywords: Revocable identity-based encryption, static assumptions, asymmetric pairings.

1 Introduction

Identity-Based Encryption (IBE) scheme is a public key cryptosystem enabling one to use arbitrary bit-string as her/his public key. In dynamic cryptosystems, user registration and revocation are important functionalities. When Boneh and Franklin proposed the first realization of IBE [BF01], they already explained how to revoke corrupted users; for an identity I of a non-revoked user at time T, $\mathtt{I}\|T$ is regarded as the identity, and Key Generation Center (KGC) issues a secret key

*The preliminary version will appear in CT-RSA 2017 [WES17]. This is the full version.

†The first author is supported by JSPS Research Fellowships for Young Scientists.

for $\mathtt{I}\|T$ to a non-revoked user $\mathtt{I}$ for each time period. Even though this simple identity-encoding method can successfully revoke users from the system, KGC's huge overhead (linear computational complexity in the number of users per each time period) is an inherent problem. To resolve this problem, Boldyreva, Goyal, and Kumar [BGK08] proposed a scalable revocation method by using the symmetric key broadcast encryption technique, so-called the Complete Subtree (CS) method [NNL01]. They called IBE with such the efficient revocation Revocable IBE (RIBE).

After the seminal work by Boldyreva, Goyal, and Kumar [BGK08], several RIBE schemes have been proposed so far. Almost all such subsequent works basically follow Boldyreva et al.'s revocation methodology. Let us briefly explain Boldyreva et al.'s approach; as in IBE, each user has a (long-term) secret key $sk_{\mathtt{I}}$. At each time T, KGC broadcasts key update information ku_T which is constructed by the CS method. Remark that no secure channel is required to send ku_T to users. A user can compute a decryption key $dk_{\mathtt{I},T}$ from ku_T and own $sk_{\mathtt{I}}$ if the user is not revoked at T. Due to the CS method, the size of ku_T is $O(r\log(n/r))$, where n is the number of maximum users and r is the number of revoked users. Thus, Bolyreva et al. RIBE scheme is scalable. The first adaptively secure RIBE scheme was proposed by Libert and Vergnaud [LV09]. Seo and Emura extended the Boldyreva et al.'s security notion to consider more practical threats; *decryption key exposure resistance (DKER)* [SE13b, SE14b]. Intuitively, this notion considers the case where several decryption keys $dk_{\mathtt{I}^*,T}$ for the target identity $\mathtt{I}^*$ are leaked to an adversary but the target decryption key $dk_{\mathtt{I}^*,T^*}$ is not exposed. This notion is important where the secret key is stored in physically secure devices such as USB pen drives to be isolated from the Internet but decryption keys are stored in weaker device such as a smart phone. They also proposed the first scalable RIBE scheme with adaptive security with DKER. The Seo-Emura RIBE is based on the Waters IBE [Wat05], so that long public parameters are inevitable. Since there exist several efficient IBE schemes, it is quite natural to ask

whether we attain an adaptively secure RIBE scheme with DKER, which achieves similar performance to efficient IBE schemes, in particular, short public parameters in prime-order groups.

Although several RIBE schemes are proposed so far [CLL$^+$12a, CLL$^+$12b, CZ15, LLP14, IWS15, PLL15, SE14c, SLLW14], none of them achieves adaptive security against decryption key exposure and short parameters (in the sense of constant public parameters and prime-order groups) at the same time. We found that the answer is not trivial due to the following reasons. Basically, there are two approaches to achieving constant-size public parameter IBE: One is to use strong assumptions such as static ones in composite-order groups and q-type ones (e.g., [Gen06, Wee16]); and the other is to apply the dual system encryption methodology [Wat09] in either prime-order or composite-order groups. Therefore, if we want to realize an RIBE scheme with constant-size public parameter under static assumptions in prime-order groups, it is quite natural to apply the latter approach for our purpose.

Unfortunately, there exists a subtle obstacle in applying the dual system encryption methodology for adaptive security with DKER. In fact, Lee observed such an obstacle [Lee16] and also, basing on his observation, pointed out a flaw of an Revocable Hierarchical IBE (RHIBE) scheme [SE15a]. Let us briefly review such an obstacle. In the dual system encryption framework, ciphertexts and secret keys can be transformed into semi-functional ones. Normal ciphertexts can be decrypted with either a normal or semi-functional key, whereas semi-functional ciphertexts can be decrypted with only a normal secret key. In the security proof, a normal challenge ciphertext and secret keys are transformed into their semi-functional forms one by one. In the process of changing some normal key (called a *target key*) into its semi-functional form, a simulator has to embed some function f into public parameters. Thus, the simulator can generate randomness $r_C := f(\mathtt{I}^*)$ for the challenge ciphertext, as well as randomness $r_K := f(\mathtt{I})$ for the target key, where $\mathtt{I}^*$ is the target identity and

$\mathtt{I}$ is an identity such that $\mathtt{I} \neq \mathtt{I}^*$. The proof goes well since f is a pairwise independent function and $\mathtt{I} \neq \mathtt{I}^*$, i.e., r_C is independent of r_K from an adversarial view in the information-theoretic sense. To the best of our knowledge, such a pairwise independent function f is necessary for proving security of all of the currently-known IBE schemes using the dual system encryption methodology. On the other hand, an adversary against the security game of RIBE can get not only a challenge ciphertext for $\mathtt{I}^*$ but also a secret key for $\mathtt{I}^*$ (see Definition 1). Therefore, we cannot argue that randomness r_C for the challenge ciphertext and randomness r_K for the secret key are independent of each other from the view point of the adversary, since it holds $r_C = r_K = f(\mathtt{I}^*)$.

Lee [Lee16] introduced a way to circumvent the above obstacle and also proposed provably secure RHIBE scheme in the adaptive adversary model. Since we can consider a 1-level HIBE as an IBE scheme, Lee's RHIBE can be considered as an adaptively secure RIBE with DKER and short public parameters. We note that, however, his approach essentially used composite-order bilinear groups. Moreover, there are other RHIBE schemes [ESY16, LP16, RLPL16, SE13a, SE14a, SE15b, SE16], but none of them satisfies both adaptive security with DKER and short parameters (i.e., short public parameters in prime-order groups) at the same time. Therefore, designing an adaptively secure RIBE scheme with DKER and short parameters (possibly through the dual system encryption approach) is still open.

1.1 Our Contribution

In this paper, we propose the first adaptively secure RIBE scheme with constant size public parameters in asymmetric bilinear groups of prime order. Our RIBE scheme also supports DKER. The security of our scheme is proved under static assumptions, which are mild variants of the symmetric external Diffie-Hellman (SXDH) assumption.

We overcome the difficulty mentioned above by the following strategy: Taking the Seo-Emura approach [SE13b]. Seo and Emura proposed an adaptively secure RIBE scheme based on the Waters IBE [Wat05], and showed a security reduction from the Waters IBE to their RIBE scheme. Note that the Waters IBE does not use the dual system encryption methodology, and requires long public parameters which depend on the bit-length of identities. Therefore, by taking the Seo-Emura approach we want to avoid the randomness correlation problem specific to dual system encryption-based RIBE schemes. Namely, we want to make a security reduction from some IBE scheme using the dual system encryption methodology to our RIBE scheme. However, the Seo-Emura technique essentially requires *the secret-key re-randomization*[1] of the underlying IBE scheme, but almost all of the dual system encryption-based IBE schemes in prime-order groups (e.g., [Wat09, Lew12, CLL$^+$14]) do not have this property.

Therefore, we employ the Jutla-Roy IBE [JR13] (and its variant [RS14]) as a promising candidate of our basic IBE scheme since it allows one to publicly re-randomize the secret key. However, the public parameter of the Jutla-Roy IBE lacks some important elements for simulating secret keys in the security proof. In the security proof taking the Seo-Emura approach, a simulator extracts the master key of the underlying IBE scheme by using the Boneh-Boyen technique [BB04], and creates a secret key $sk_{\mathtt{I}^*}$ or decryption key $dk_{\mathtt{I}^*,T}$ for any T, where $\mathtt{I}^*$ is the challenge identity and T is a time period such that it is not the challenge one. The Boneh-Boyen technique requires some group elements that contain the master key in the exponent in the public parameter of the underlying IBE, however the original Jutla-Roy IBE does not contain them (For details, see Section 3). Hence, we modify the Jutla-Roy IBE so that the Seo-Emura technique can be applied to it, and we prove the security under the Augmented Decisional Diffie-Hellman on $\mathbb{G}_1$ (ADDH1), which is a new static

[1] It means that each secret key can be re-randomized with fresh randomness.

Table 1: Efficiency Comparison among adaptively secure RIBE schemes with DKER.

Scheme	$\#mpk$	$\#msk$	$\#C$
Seo-Emura [SE13b, SE14b]	$(6+\ell)\lvert\mathbb{G}_p\rvert$	$\lvert\mathbb{G}_p\rvert$	$3\lvert\mathbb{G}_p\rvert + \lvert\mathbb{G}_T^{\mathsf{sym}}\rvert$
Lee [Lee16] ($L=1$)	$8\lvert\mathbb{G}_N\rvert + 3\lvert\mathbb{G}_T^{\mathsf{comp}}\rvert$	$\lvert\mathbb{G}_N\rvert$	$4\lvert\mathbb{G}_N\rvert + \lvert\mathbb{G}_T^{\mathsf{comp}}\rvert$
Proposed Scheme	$7\lvert\mathbb{G}_1\rvert + 11\lvert\mathbb{G}_2\rvert + \lvert\mathbb{G}_T^{\mathsf{asym}}\rvert$	$2\lvert\mathbb{G}_2\rvert$	$4\lvert\mathbb{G}_1\rvert + \lvert\mathbb{G}_T^{\mathsf{asym}}\rvert + \lvert\mathbb{Z}_p\rvert$

Scheme	$\#sk$	$\#ku$	$\#dk$	Assumption
Seo-Emura [SE13b, SE14b]	$(2\log n)\lvert\mathbb{G}_p\rvert$	$(2r\log(n/r))\lvert\mathbb{G}_p\rvert$	$3\lvert\mathbb{G}_p\rvert$	DBDH
Lee [Lee16] ($L=1$)	$(2\log n)\lvert\mathbb{G}_N\rvert$	$(2r\log(n/r))\lvert\mathbb{G}_N\rvert + 2\lvert\mathbb{Z}_N\rvert$	$4\lvert\mathbb{G}_N\rvert$	Static
Proposed Scheme	$(5\log n)\lvert\mathbb{G}_2\rvert$	$(3r\log(n/r))\lvert\mathbb{G}_2\rvert$	$6\lvert\mathbb{G}_2\rvert$	ADDH1, DDH2

Let $|\mathbb{G}_1|$, $|\mathbb{G}_2|$, and $|\mathbb{G}_T^{\mathsf{asym}}|$ be the bit-length of an element of asymmetric bilinear groups $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ respectively. Let $|\mathbb{G}_p|$ and $|\mathbb{G}_T^{\mathsf{sym}}|$ be the bit-length of an element of symmetric bilinear groups $\mathbb{G}_p$ and $\mathbb{G}_T$ employed in [SE13b, SE14b], respectively. Let $|\mathbb{G}_N|$ and $|\mathbb{G}_T^{\mathsf{comp}}|$ be the bit-length of an element of symmetric bilinear groups $\mathbb{G}_N$ and $\mathbb{G}_T$ of composite order $N = p_1p_2p_3$, where p_1, p_2, and p_3 are distinct prime numbers, employed in [Lee16], respectively. Let $|\mathbb{Z}_p|$ and $|\mathbb{Z}_N|$ be the bit-length of an element of $\mathbb{Z}_p$ and $\mathbb{Z}_N$, respectively. On 256-bit Barreto-Naehrig curve [BN06], $|\mathbb{G}_1| = 256$, $|\mathbb{G}_2| = |\mathbb{G}_p| = 512$, and $|\mathbb{G}_T^{\mathsf{asym}}| = |\mathbb{G}_T^{\mathsf{sym}}| = 3072$ due to [CLL$^+$14]. Note that $|\mathbb{G}_N|$ and $|\mathbb{G}_T^{\mathsf{comp}}|$ should be much larger so that N cannot be factored. L is the hierarchy depth, n is the maximum number of users, r is the number of revoked users, and ℓ is the bit-length of identity. For example, if 32 byte e-mail address is regarded as identity, then $\ell = 256$.

assumption, and Decisional Diffie-Hellman on $\mathbb{G}_2$ (DDH2) assumptions. The ADDH1 assumption is newly introduced in this paper, and therefore it is a non-standard one. However, this assumption is not so complicated and similar to the previously used assumption in [RCS12a]. The security of the ADDH1 assumption is proved in the generic bilinear group model.

We then propose an RIBE scheme based on the Jutla-Roy IBE, and the security is proved by making a security reduction from the modified Jutla-Roy IBE to the RIBE scheme.[2] As a result, we obtain the first RIBE scheme that achieves adaptive security with DKER and constant-size public parameters in prime-order asymmetric bilinear groups. Furthermore, our proof technique provides a better reduction loss, which is elaborated in the next paragraph.

Efficiency Comparison: We give an efficiency comparison in Table 1. All of the schemes meet adaptive security with DKER. We use the KUNode algorithm for efficient revocation as in previous RIBE schemes (For details, see Section 2 or [NNL01]). Therefore, the sizes of secret keys and key updates in every scheme are $O(\log n)$ and $O(r\log(n/r))$, respectively, due to the KUNode algorithm. Lee's scheme [Lee16] is less efficient than the others since it is constructed over composite-order bilinear groups. Our scheme is more efficient than the Seo-Emura RIBE in terms of constant-size public parameters and asymmetric pairings, and other parameters are comparable to those of the Seo-Emura RIBE. In addition, our proof technique provides a better reduction loss than that of the Seo-Emura RIBE. More precisely, the reduction loss of our scheme is $O(q_1q|\mathcal{T}|)$, whereas that of the Seo-Emura RIBE is $O(\ell q^2|\mathcal{T}|)$, where ℓ is the bit-length of identity, q is the maximum number of queries in the security game, q_1 is the maximum number of queries *before the challenge phase* in

[2] This situation is the same as that of Ishida et al.'s construction [IWS15]. Since the Kiltz-Galindo IB-KEM [KG09] is not directly applicable due to the same reason, they constructed a variant of the Kiltz-Galindo IB-KEM, and then showed a security reduction from the variant scheme to their scheme.

the security game, and $|\mathcal{T}|$ is the number of time periods in the schemes.

1.2 Paper Organization

In Section 2, we describe notation and definitions throughout this paper. In Section 3, we propose an IBE scheme, which is used as the underlying IBE scheme of our RIBE scheme, based on the Jutla-Roy IBE. In Section 4, we show the first adaptively secure RIBE scheme with DKER and short public parameters in prime-order groups, and some extensions are discussed in Section 5. We finally conclude in Section 6.

2 Preliminaries

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

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

A bilinear map e is called symmetric or a "Type-1" pairing if $\mathbb{G}_1 = \mathbb{G}_2$. Otherwise, it is called asymmetric. In the asymmetric setting, e is called a "Type-2" pairing if there is an efficiently computable isomorphism from $\mathbb{G}_2$ to $\mathbb{G}_1$. If no efficiently computable isomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$ is known, then it is called a "Type-3" pairing. Throughout this paper, we focus on the Type-3 pairing. Type-3 is the most efficient setting since compared to Type-1, the size of representation of $\mathbb{G}_1$ in the Type-3 setting is smaller and whole operations in the Type-3 setting are more efficient; and compared to Type-2, the size of representation of $\mathbb{G}_2$ in the Type-3 setting is smaller and group operations in $\mathbb{G}_2$ in the Type-3 are more efficient. For details, see [GPS08].

KUNode Algorithm. To reduce costs of a revocation process, we use a binary tree structure and apply the following KUNode algorithm as in the previous RIBE schemes [BGK08, LV09, SE13b]. $\mathsf{KUNode}(\mathsf{BT}, RL, T)$ takes as input a binary tree BT, a revocation list RL, and a time period $T \in \mathcal{T}$, and outputs a set of nodes. When η is a non-leaf node, then we write η_L and η_R as the left and right child of η, respectively. When η is a leaf node, $\mathsf{Path}(\mathsf{BT}, \eta)$ denotes the set of nodes on the path from η to the *root*. Each user is assigned to a leaf node. If a user who is assigned to η is revoked on a time period $T \in \mathcal{T}$, then $(\eta, T) \in RL$. $\mathsf{KUNode}(\mathsf{BT}, RL, T)$ is executed as follows. It sets $\mathcal{X} := \emptyset$ and $\mathcal{Y} := \emptyset$. For any $(\eta_i, T_i) \in RL$, if $T_i \leq T$ then it adds $\mathsf{Path}(\mathsf{BT}, \eta_i)$ to $\mathcal{X}$ (i.e., $\mathcal{X} := \mathcal{X} \cup \mathsf{Path}(\mathsf{BT}, \eta_i)$). That is, KUNode adds at most $r \log n$ nodes to $\mathcal{X}$ where $r = |RL|$ and n is the number of leaves of BT. Then, for any $\eta \in \mathcal{X}$, if $\eta_L \notin \mathcal{X}$, then it adds η_L to $\mathcal{Y}$. If $\eta_R \notin \mathcal{X}$, then it adds η_R to $\mathcal{Y}$. That is, KUNode adds at most $r \log n$ nodes to $\mathcal{Y}$. Actually, due to the result of [NNL01], the size of $\mathcal{Y}$ is $O(r \log(n/r))$, and the time complexity is $O(\log \log n)$. Finally, it outputs $\mathcal{Y}$ if $\mathcal{Y} \neq \emptyset$. If $\mathcal{Y} = \emptyset$, then it adds *root* to $\mathcal{Y}$ and outputs $\mathcal{Y}$.

Revocable Identity-based Encryption. An RIBE scheme Π consists of seven-tuple algorithms (Setup, SKGen, KeyUp, DKGen, Enc, Dec, Revoke) defined as follows: For simplicity, we omit a public parameter in the input of all algorithms except for the Setup algorithm.

- $(mpk, msk, RL, st) \leftarrow \mathsf{Setup}(\lambda, N)$: A probabilistic algorithm for setup. It takes a security parameter λ and the maximum number of users N as input and outputs a public parameter mpk, a master secret key msk, an initial revocation list $RL = \emptyset$ and a state st.
- $(sk_{\mathtt{I}}, st) \leftarrow \mathsf{SKGen}(st, \mathtt{I})$: An algorithm for private key generation. It takes st and an identity $\mathtt{I} \in \mathcal{I}$ as input and outputs a secret key $sk_{\mathtt{I}}$ and updated state information st.[3]
- $ku_T \leftarrow \mathsf{KeyUp}(msk, st, RL, T)$: An algorithm for key update generation. It takes msk, state st, a current revocation list RL, and a time period T as input, and then outputs key update ku_T.
- $dk_{\mathtt{I},T}$ or $\perp \leftarrow \mathsf{DKGen}(sk_{\mathtt{I}}, ku_T)$: A probabilistic algorithm for decryption key generation. It takes $sk_{\mathtt{I}}$ and ku_T as input and then outputs a decryption key $dk_{\mathtt{I},T}$ at T or $\perp$ if $\mathtt{I}$ has been revoked by T.
- $C_{\mathtt{I},T} \leftarrow \mathsf{Enc}(M, \mathtt{I}, T)$: A probabilistic algorithm for encryption. It takes $M \in \mathcal{M}$, $\mathtt{I} \in \mathcal{I}$, and $T \in \mathcal{T}$ as input and then outputs a ciphertext $C_{\mathtt{I},T}$.
- M or $\perp \leftarrow \mathsf{Dec}(dk_{\mathtt{I},T}, C_{\mathtt{I},T})$: A deterministic algorithm for decryption. It takes $dk_{\mathtt{I},T}$ and $C_{\mathtt{I},T}$ as input and then outputs M or $\perp$.
- $RL \leftarrow \mathsf{Revoke}(\mathtt{I}, T, RL, st)$: An algorithm for revocation. It takes $(\mathtt{I}, T) \in \mathcal{I} \times \mathcal{T}$, the current revocation list RL, and a state st as input and then outputs an updated revocation list RL.

In the above model, we assume that Π meets the following correctness property: For all security parameter $\lambda \in \mathbb{N}$, all $(mpk, msk, RL, st) \leftarrow \mathsf{Setup}(\lambda, N)$, all $M \in \mathcal{M}$, all $\mathtt{I} \in \mathcal{I}$, all $T \in \mathcal{T}$, if $\mathtt{I}$ is not revoked on $T \in \mathcal{T}$, it holds that $M = \mathsf{Dec}(\mathsf{DKGen}(\mathsf{SKGen}(st, \mathtt{I}), \mathsf{KeyUp}(msk, st, RL, T)), \mathsf{Enc}(M, \mathtt{I}, T))$.

We describe the notion of indistinguishability against chosen plaintext attack (IND-RID-CPA). Note that this notion also captures DKER, which was introduced by Seo and Emura [SE13b], and this security model is the strongest known one. Let $\mathcal{A}$ be a PPT adversary, and $\mathcal{A}$'s advantage against IND-RID-CPA security is defined by

$$Adv_{\Pi,\mathcal{A}}^{\textit{IND-RID-CPA}}(\lambda, N) := \left| \Pr\left[b' = b \;\middle|\; \begin{array}{l} (mpk, msk, RL, st) \leftarrow \mathsf{Setup}(\lambda, N), \\ (M_0^*, M_1^*, \mathtt{I}^*, T^*, state) \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{find}, mpk), \\ b \xleftarrow{\$} \{0,1\}, \\ C_{\mathtt{I}^*,T^*}^* \leftarrow \mathsf{Enc}(M_b^*, \mathtt{I}^*, T^*), \\ b' \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{guess}, C_{\mathtt{I}^*,T^*}^*, state) \end{array} \right] - \frac{1}{2} \right|.$$

Here, $\mathcal{O}$ is a set of oracles $\{SKGen(\cdot), KeyUp(\cdot), Revoke(\cdot,\cdot), DKGen(\cdot,\cdot)\}$ defined as follows.

SKGen$(\cdot)$**:** For a query $\mathtt{I} \in \mathcal{I}$, it stores and returns $\mathsf{SKGen}(st, \mathtt{I})$.

KeyUp$(\cdot)$**:** For a query $T \in \mathcal{T}$, it stores and returns $\mathsf{KeyUp}(msk, RL, st, T)$.

Revoke$(\cdot,\cdot)$**:** For a query $(\mathtt{I}, T) \in \mathcal{I} \times \mathcal{T}$, it updates a revocation list RL by running $\mathsf{Revoke}(\mathtt{I}, T, RL, st)$.

[3] We consider the SKGen algorithm in the sense of history-free RHIBE [SE15b, SE16], i.e., the algorithm takes st, rather than msk, as input.

DKGen$(\cdot,\cdot)$**:** For a query $(\mathtt{I},T) \in \mathcal{I} \times \mathcal{T}$, it finds $sk_{\mathtt{I}}$ and ku_T generated by the *SKGen* and *KeyUp* oracles, respectively (If $sk_{\mathtt{I}}$ has not been generated yet, *DKGen* executes $(sk_{\mathtt{I}}, st) \leftarrow \mathsf{SKGen}(st, \mathtt{I})$).[4] *DKGen* returns $\mathsf{DKGen}(sk_{\mathtt{I}}, ku_T)$ and stores it unless it is $\perp$.

The above oracles represent the following realistic threats and situations: *SKGen* represents the collusion among users as in ordinary IBE. $\mathcal{A}$ can access *KeyUp* since key updates are broadcasted by the KGC. The reason why $\mathcal{A}$ can access *Revoke* is an RIBE scheme should be secure against any situations in terms of the revocation list. *DKGen* represents decryption key exposure.

We then impose the following restrictions on $\mathcal{A}$. Specifically, the first three restrictions are placed to take into account practical situations, and we circumvent some trivial attacks by the other restrictions.

1. *KeyUp*$(\cdot)$ and *Revoke*$(\cdot,\cdot)$ can be queried at a time period which is later than or equal to that of all previous queries.

2. *Revoke*$(\cdot,\cdot)$ cannot be queried at a time period T after issuing T to $\mathsf{KeyUp}(\cdot)$.

3. *DKGen*$(\cdot,\cdot)$ cannot be queried at T before issuing T to $\mathsf{KeyUp}(\cdot)$.

4. If $\mathtt{I}^*$ was issued to *SKGen*$(\cdot)$ at T', then $(\mathtt{I}^*, T)$ must be issued to $\mathsf{Revoke}(\cdot,\cdot)$ such that $T' \le T \le T^*$.

5. $(\mathtt{I}^*, T^*)$ cannot be issued to *DKGen*$(\cdot,\cdot)$.

Definition 1. *An RIBE scheme* Π *is said to be IND-RID-CPA secure if for all PPT adversaries* $\mathcal{A}$*,* $Adv_{\Pi,\mathcal{A}}^{\text{IND-RID-CPA}}(\lambda, N)$ *is negligible in* λ*.*

3 The Basic IBE Scheme

We begin with reviewing Seo and Emura's approach for transforming IBE to RIBE [SE13b]. Although their approach is not generic, it seems quite broadly applicable to the other IBE schemes. We find some requirements for applying their technique. Then, we propose an IBE scheme satisfying such the requirements, which has short public parameters and over prime-order bilinear groups.

Seo and Emura constructed an RIBE scheme based on the Waters IBE [Wat05] and provided a security reduction to the Waters IBE. In the reduction, almost all queries can be easily simulated due to the adaptive security of the underlying IBE. The most non-trivial part in the reduction is simulating decryption keys for $(\mathtt{I}^*, T)$, where $\mathtt{I}^*$ is the target identity, since the security of usual IBE scheme does not handle this case related to $\mathtt{I}^*$. To this end, Seo and Emura employed two techniques; the Boneh-Boyen technique [BB04] and secret-key *re-randomization*.

The Boneh-Boyen technique is originally for *selectively secure* scheme[5]; that is, if the simulator knows the target (time T^* in our case) in advance, then the simulator embeds it into public parameters so that the simulator can simulate all the other queries not related to T^*.[6] The Boneh-Boyen technique enables the simulator to compute decryption keys for $(\mathtt{I}^*, T)$ with biased distribution, where T is not the target time. The secret-key re-randomization can resolve the biased distribution by forcing that all decryption keys have uniform randomness.

[4] Contrary to $sk_{\mathtt{I}}$, ku_T is already stored by the *KeyUp* oracle due to the restrictions on the oracles.

[5] Although our goal is adaptive security, the polynomial reduction loss enables one to use the selective security technique in terms of (polynomial-size) time period.

[6] Although the decryption key $(\mathtt{I}^*, T)$ is related to the target identity $\mathtt{I}^*$, it is not related to T^* so that the Boneh-Boyen technique is applicable.

From the above interpretation, we find two requirements for the input IBE; (1) the secret-key re-randomization property and (2) applicability of the Boneh-Boyen technique. The latter requirement can be further segmentalized. (2-1) Each component of a secret key contains at most one component of a master key and (2-2) each component of the master-key is available in the public parameters in some form of elements in source-groups (of bilinear groups). The former is due to that the Boneh-Boyen technique can extract at most one master-key component from each secret-key component. The latter is due to that in the security reduction the master-key is embedded into key updates that consist of only elements in source-groups by using the master-key-related public parameters.[7]

The Waters IBE satisfies all the above requirements, but most of dual-system-encryption-based IBE schemes in prime-order groups do not. For example, the first scheme by Waters [Wat09] and almost all of the IBE schemes using dual pairing vector spaces (DPVS) (e.g.,[Lew12, CLL$^+$14]) do not satisfy any requirement, in particular, the public re-randomization requirement.

3.1 Modified Jutla-Roy IBE

We employ a modified version of the Jutla-Roy IBE [JR13] (and its variant [RS14]). The original scheme satisfies two requirements (1) and (2-1). In this subsection, we modify the Jutla-Roy IBE to additionally satisfy the requirement (2-2).

The master key of the Jutla-Roy IBE is $(y_0, x_0) \in \mathbb{Z}_p^2$. To get a basic IBE scheme for our RIBE scheme based on the Jutla-Roy IBE, we add the master key in the forms of elements in $\mathbb{G}_1$ and $\mathbb{G}_2$ with a random mask $\beta \in \mathbb{Z}_p^\times$, respectively, to the public parameters. Specifically, we add four group elements $(\chi_1 := g_1^{\beta(-x_0\alpha+y_0)}, g_2^{x_0\beta}, g_2^{y_0\beta}, g_2^{1/\beta})$ to the original public parameter. However, we then cannot apply the original security proof of the Jutla-Roy IBE, and so we add a new twist to the proof. The modified Jutla-Roy IBE Π_{JR} =(Init, KeyGen, IBEnc, IBDec) is constructed as follows.[8]

- $\mathsf{Init}(\lambda)$: It runs $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, g_1, g_2, e) \leftarrow \mathcal{G}$. It chooses $x_0, y_0, x_1, y_1, x_2, y_2, x_3, y_3 \xleftarrow{\$} \mathbb{Z}_p$ and $\alpha, \beta \xleftarrow{\$} \mathbb{Z}_p^\times$, and sets

  $$z = e(g_1, g_2)^{-x_0\alpha+y_0}, \quad u_1 := g_1^{-x_1\alpha+y_1}, \quad w_1 := g_1^{-x_2\alpha+y_2}, \quad h_1 := g_1^{-x_3\alpha+y_3}, \quad \chi_1 := g_1^{\beta(-x_0\alpha+y_0)}.$$

  It outputs

  $$PP := (g_1, g_1^\alpha, u_1, w_1, h_1, \chi_1, g_2, g_2^{x_1}, g_2^{x_2}, g_2^{x_3}, g_2^{y_1}, g_2^{y_2}, g_2^{y_3}, z, g_2^{x_0\beta}, g_2^{y_0\beta}, g_2^{\frac{1}{\beta}}),$$
  $$MK := (g_2^{y_0}, g_2^{-x_0}).$$

- $\mathsf{KeyGen}(PP, MK, \mathtt{I})$: Parse MK as (d_1', d_2'). It chooses $r \xleftarrow{\$} \mathbb{Z}_p$ and computes

  $$D_1 := (g_2^{y_2})^r, \quad D_1' := d_1'\Big((g_2^{y_1})^{\mathtt{I}} g_2^{y_3}\Big)^r,$$
  $$D_2 := (g_2^{x_2})^{-r}, \quad D_2' := d_2'\Big((g_2^{x_1})^{\mathtt{I}} g_2^{x_3}\Big)^{-r}, \quad D_3 := g_2^r.$$

  It outputs $SK_{\mathtt{I}} := (D_1, D_1', D_2, D_2', D_3)$.

[7] In (usual-but-not-all) pairing-based IBE schemes, private keys consist of elements in source-groups. Since both key updates and secret keys of RIBE are materials for decryption keys, they also should consist of source-group elements.

[8] The syntax of IBE is given in Appendix A.

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

$$C_0 := Mz^t,\ C_1 := g_1^t,\ C_2 := (g_1^{\alpha})^t,\ C_3 := \left(u_1^{\mathtt{I}} w_1^{\mathtt{tag}} h_1\right)^t.$$

  It outputs $C := (C_0, C_1, C_2, C_3, \mathtt{tag})$.

- $\mathsf{IBDec}(PP, SK_{\mathtt{I}}, C)$: Parse $SK_{\mathtt{I}}$ and C as $(D_1, D_1', D_2, D_2', D_3)$ and $(C_0, C_1, C_2,\ C_3, \mathtt{tag})$, respectively. It computes

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

We show the correctness of Π_{JR}. Suppose that $sk_{\mathtt{I}} = (D_1, D_1', D_2, D_2', D_3)$ and $C = (C_0, C_1, C_2, C_3,$ $)$ are correctly generated. Then, we have

$$\begin{aligned}
&\frac{C_0 e(C_3, D_3)}{e(C_1, D_1^{\mathtt{tag}} D_1') e(C_2, D_2^{\mathtt{tag}} D_2')} \\
&\qquad = Me(g_1, g_2)^{(-x_0\alpha + y_0)t} \frac{e(g_1^{t(\mathtt{I}(-x_1\alpha+y_1)+\mathtt{tag}(-x_2\alpha+y_2)-x_3\alpha+y_3)}, g_2^r)}{e(g_1^t, g_2^{y_2 r \mathtt{tag} + y_0 + r(y_1\mathtt{I}+y_3)}) e(g_1^{\alpha t}, g_2^{-x_2 r \mathtt{tag} - x_0 - r(x_1\mathtt{I}+x_3)})} \\
&\qquad = Me(g_1, g_2)^{(-x_0\alpha + y_0)t} \frac{1}{e(g_1^t, g_2^{y_0}) e(g_1^{\alpha t}, g_2^{-x_0})} = M.
\end{aligned}$$

3.2 Proof of Security

We describe complexity assumptions used for proving the security proof of the modified Jutla-Roy IBE.

First, we give the definition of the decisional Diffie-Hellman (DDH) assumption in $\mathbb{G}_1$ and $\mathbb{G}_2$, which are called the DDH1 and DDH2 assumptions, respectively. We say that the SXDH assumption holds if both the DDH1 and DDH2 assumptions hold. Let $\mathcal{A}$ be a PPT adversary and we consider $\mathcal{A}$'s advantage against the DDHi problem ($i = 1, 2$) as follows.

$$Adv_{\mathcal{G},\mathcal{A}}^{DDHi}(\lambda) := \left| \Pr\left[b' = b \;\middle|\; \begin{array}{l} D := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e) \leftarrow \mathcal{G}, \\ c_1, c_2 \xleftarrow{\$} \mathbb{Z}_p,\ b \xleftarrow{\$} \{0,1\}, \\ \text{if } b = 0 \text{ then } Z := g_i^{c_1 c_2},\ \text{else } Z \xleftarrow{\$} \mathbb{G}_i, \\ b' \leftarrow \mathcal{A}(\lambda, D, g_i^{c_1}, g_i^{c_2}, Z) \end{array} \right] - \frac{1}{2} \right|.$$

Definition 2 (DDHi Assumption)**.** *The DDHi assumption relative to a generator $\mathcal{G}$ holds if for all PPT adversaries $\mathcal{A}$, $Adv_{\mathcal{G},\mathcal{A}}^{DDHi}(\lambda)$ is negligible in λ.*

Definition 3 (SXDH Assumption)**.** *We say that the symmetric external Diffie-Hellman (SXDH) assumption relative to a generator $\mathcal{G}$ holds if both the DDH1 and DDH2 assumptions relative to $\mathcal{G}$ hold.*

We then introduce a new assumption based on the DDH1 assumption, which is called *Augmented DDH1 (ADDH1) assumption.* Let $\mathcal{A}$ be a PPT adversary and we consider $\mathcal{A}$'s advantage against the ADDH1 problem as follows.

$$Adv_{\mathcal{G},\mathcal{A}}^{ADDH1}(\lambda) := \left| \Pr\left[b' = b \;\middle|\; \begin{array}{l} D := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e) \leftarrow \mathcal{G}(\lambda), \\ d, c_1, c_2 \xleftarrow{\$} \mathbb{Z}_p, c_3 \xleftarrow{\$} \mathbb{Z}_p^{\times},\ b \xleftarrow{\$} \{0,1\}, \\ \text{if } b = 0 \text{ then } Z := g_1^{c_1 c_2},\ \text{else } Z \xleftarrow{\$} \mathbb{G}_1, \\ b' \leftarrow \mathcal{A}(\lambda, D, g_1^{c_1}, g_1^{c_2}, g_1^{dc_3}, g_2^{d}, g_2^{c_2 c_3}, g_2^{dc_3}, g_2^{\frac{1}{c_3}}, Z) \end{array} \right] - \frac{1}{2} \right|.$$

Definition 4 (ADDH1 Assumption)**.** *The ADDH1 assumption relative to a generator $\mathcal{G}$ holds if for all PPT adversaries $\mathcal{A}$, $Adv^{ADDH1}_{\mathcal{G},\mathcal{A}}(\lambda)$ is negligible in λ.*

This assumption is similar to the DDH2v assumption ("v" stands for "variant"), which was used for constructing the Lewko-Waters IBE [LW10] in prime-order groups in [RCS12a]. Similarly, we can also consider the DDH1v assumption.[9] The authors of [RCS12a] argued that the DDH2v (resp., DDH1v) assumption is the minimal assumption when one tries to put some information about c_1 or c_2 in an instance of DDH1 (resp., DDH2) while staying in the hardness of the problem. We define the ADDH1 problem by removing g_1^d from the DDH1v problem and adding $g_1^{dc_3}$ and g_2^{1/c_3}. Therefore, we may say this new assumption is also a not-so-strange one. Actually, we prove the security of this assumption in the generic bilinear group model as follows (For the formal proof, see Appendix B).

Theorem 1 (Informal)**.** *Let $\mathcal{A}$ be an algorithm that attempts to solve the ADDH1 problem in the generic group model. $\mathcal{A}$ makes at most q queries to the oracles computing the group actions in $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$, and the bilinear map e. Then, the advantage ϵ of $\mathcal{A}$ in solving the problem is bounded by $\epsilon \leq 3(q+11)^2/4p$.*

We prove the security of Π_{JR} under the above assumptions.

Theorem 2. *If the ADDH1 and DDH2 assumptions hold, then the resulting Jutla-Roy IBE Π_{JR} is IND-ID-CPA secure.*

Proof. Our security proof is the same as that of the Jutla-Roy IBE except that we have to care the extra terms $(\chi_1, g_2^{x_0\beta}, g_2^{y_0\beta}, g_2^{1/\beta})$ that were added to their scheme. We replace the DDH1 assumption of Jutla-Roy's proof with "DDH1 with the additional instance", the ADDH1 assumption, in order to treat these extra terms. More specifically, we need the ADDH1 assumption in the proof of indistinguishability of the semi-functional challenge ciphertext and the random element in the ciphertext space (see Lemma 3 for details).

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

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

$$\tilde{C}_0 := C_0 e(g_1, g_2)^{-x_0\mu} = Me(g_1, g_2)^{-x_0(\alpha t+\mu)+y_0 t},$$
$$\tilde{C}_1 := C_1,$$
$$\tilde{C}_2 := C_2 g_1^{\mu} = g_1^{\alpha t+\mu},$$
$$\tilde{C}_3 := C_3\Big((g_1^{x_1})^{\mathtt{I}}(g_1^{x_2})^{\mathtt{tag}} g_1^{x_3}\Big)^{-\mu} = C_3 g_1^{-\mu(x_1\mathtt{I}+x_2\mathtt{tag}+x_3)} = g_1^{-(\alpha t+\mu)(x_1\mathtt{I}+x_2\mathtt{tag}+x_3)} g_1^{t(y_1\mathtt{I}+y_2\mathtt{tag}+y_3)},$$

and $\widetilde{\mathtt{tag}} := \mathtt{tag}$, where $\mu \xleftarrow{\$} \mathbb{Z}_p^{\times}$. Note that the master key $g_2^{-x_0}$ is needed to generate the semi-functional ciphertext.

Semi-functional Secret Key: Parse a normal secret key $SK_{\mathtt{I}}$ as $(D_1, D_1', D_2, D_2', D_3)$. A semi-functional secret key $\widetilde{SK}_{\mathtt{I}} := (\tilde{D}_1, \tilde{D}_1', \tilde{D}_2, \tilde{D}_2', \tilde{D}_3)$ is computed as follows:

$$\tilde{D}_1 := D_1 g_2^{\gamma} = g_2^{y_2 r+\gamma},$$

[9] We give the formal definition of the DDH2v and DDH1v assumptions in the full version of this paper.

$$\begin{aligned}
\tilde{D}'_1 &:= D'_1 g_2^{\gamma\phi} = g_2^{y_0 + r(\mathtt{I}y_1+y_3)+\gamma\phi}, \\
\tilde{D}_2 &:= D_2 g_2^{-\frac{\gamma}{\alpha}} = g_2^{-rx_2-\frac{\gamma}{\alpha}}, \\
\tilde{D}'_2 &:= D'_2 g_2^{-\frac{\gamma\phi}{\alpha}} = g_2^{-x_0-r(\mathtt{I}x_1+x_3)-\frac{\gamma\phi}{\alpha}}, \\
\tilde{D}_3 &:= D_3,
\end{aligned}$$

where $\phi \xleftarrow{\$} \mathbb{Z}_p$ and $\gamma \xleftarrow{\$} \mathbb{Z}_p^\times$. Note that in order to generate the semi-functional secret key, $g_2^{\frac{1}{\alpha}}$ is needed in addition to the public parameter.

A semi-functional ciphertext for $\mathtt{I}$ can be decrypted with a secret key for $\mathtt{I}$. This fact can be easily checked by

$$\frac{e(g_1, g_2)^{-x_0\mu} e(g_1^{-\mu(x_1\mathtt{I}+x_2\mathtt{tag}+x_3)}, D_3)}{e(g_1^{\mu}, D_2^{\mathtt{tag}} D'_2)} = 1_{\mathbb{G}_T},$$

where $1_{\mathbb{G}_T}$ is an identity element of $\mathbb{G}_T$. Also, a normal ciphertext can be decrypted with a semi-functional secret key since it holds

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

We define the following games:

Game$_{\textsf{Real}}$: This is the same as the IND-ID-CPA game.

Game$_0$: This is the same as $\textsf{Game}_{\textsf{Real}}$ except that the challenge ciphertext is semi-functional.

Game$_k$ $(1 \le k \le q)$: This is the same as $\textsf{Game}_0$ except for the following modification: Let q be the maximum number of identities issued to the *KeyGen* oracle, and $\mathtt{I}_i$ $(1 \le i \le q)$ be an i-th identity issued to the oracle. If queries regarding the first k identities $\mathtt{I}_1, \ldots, \mathtt{I}_k$ are issued, then semi-functional keys are returned. The rest of keys (i.e., keys for $\mathtt{I}_{k+1}, \ldots, \mathtt{I}_q$) are normal.

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

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

$$Adv^{\textit{IND-ID-CPA}}_{\Pi_{\text{JR}},\mathcal{A}}(\lambda) \le |S_{\textsf{Real}} - S_0| + \sum_{i=1}^{q} |S_{i-1} - S_i| + |S_q - S_{\textsf{Final}}| + \left| S_{\textsf{Final}} - \frac{1}{2} \right|.$$

The rest of the proof follows from the following lemmas.

Lemma 1. *$|S_{\textsf{Real}} - S_0| \le 2Adv^{DDH1}_{\mathcal{G},\mathcal{B}}(\lambda)$.*

Proof. At the beginning, a PPT adversary $\mathcal{B}$ receives an instance $(g_1, g_1^{c_1}, g_1^{c_2}, g_2, Z)$ of the DDH1 problem. Then, $\mathcal{B}$ randomly chooses $x_0, y_0, x_1, y_1, x_2, y_2, x_3, y_3 \xleftarrow{\$} \mathbb{Z}_p$ and $\beta \xleftarrow{\$} \mathbb{Z}_p^\times$, and creates

$$\begin{aligned}
&z := e(g_1^{c_1}, g_2)^{-x_0} e(g_1, g_2)^{y_0}, \quad u_1 := (g_1^{c_1})^{-x_1} g_1^{y_1}, \quad w_1 := (g_1^{c_1})^{-x_2} g_1^{y_2}, \\
&h_1 := (g_1^{c_1})^{-x_3} g_1^{y_3}, \quad \chi_1 := (g_1^{c_1})^{-x_0\beta} g_1^{y_0\beta}.
\end{aligned}$$

$\mathcal{B}$ sends $mpk := (g_1, g_1^{\alpha}, u_1, w_1, h_1, \chi_1, g_2, g_2^{x_1}, g_2^{x_2}, g_2^{x_3}, g_2^{y_1}, g_2^{y_2}, g_2^{y_3}, z, g_2^{\beta x_0}, g_2^{\beta y_0}, g_2^{\frac{1}{\beta}})$ to $\mathcal{A}$. Note that $\mathcal{B}$ knows a master key $msk := (g_2^{y_0}, g_2^{-x_0})$ and we implicitly set $\alpha := c_1$.

KeyGen oracle. $\mathcal{B}$ can simulate the oracle since $\mathcal{B}$ knows the master key.

Challenge. $\mathcal{B}$ receives $(M_0^*, M_1^*, \mathtt{I}^*)$ from $\mathcal{A}$. $\mathcal{B}$ chooses $d \overset{\$}{\leftarrow} \{0,1\}$. $\mathcal{B}$ chooses $\mathtt{tag}^* \overset{\$}{\leftarrow} \mathbb{Z}_p$ and computes

$$\begin{aligned}&C_0^* := M_d^* e(Z, g_2)^{-x_0} e(g_1^{c_2}, g_2)^{y_0},\ C_1^* := g_1^{c_2},\ C_2^* := Z,\\ &C_3^* := Z^{-x_1 \mathtt{I}^* - x_2 \mathtt{tag}^* - x_3} (g_1^{c_2})^{y_1 \mathtt{I}^* + y_2 \mathtt{tag}^* + y_3}.\end{aligned}$$

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

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

$$\begin{aligned}C_0^* &= M_d^* e(g_1, g_2)^{-x_0(c_1 c_2 + \mu) + y_0 c_2} = M_d^* e(g_1, g_2)^{-x_0(\alpha t + \mu) + y_0 t},\\ C_2^* &= g_1^{c_1 c_2 + \mu} = g_1^{\alpha t + \mu},\\ C_3^* &= g_1^{-(c_1 c_2 + \mu)(x_1 \mathtt{I}^* + x_2 \mathtt{tag}^* + x_3)} g_1^{c_2(y_1 \mathtt{I}^* + y_2 \mathtt{tag}^* + y_3)}\\ &= g_1^{-(\alpha t + \mu)(x_1 \mathtt{I}^* + x_2 \mathtt{tag}^* + x_3)} g_1^{t(y_1 \mathtt{I}^* + y_2 \mathtt{tag}^* + y_3)}.\end{aligned}$$

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

Lemma 2. *$|S_{k-1} - S_k| \leq 2Adv_{\mathcal{G},\mathcal{B}}^{DDH2}(\lambda)$ for every $k \in \{1, 2, \ldots, q\}$.*

Proof. At the beginning, a PPT adversary $\mathcal{B}$ receives an instance $(g_1, g_2, g_2^{c_1}, g_2^{c_2}, Z)$ of the DDH2 problem. Then, $\mathcal{B}$ randomly chooses $x_0', y_0, x_1', y_1', y_1'', x_2', x_3', y_3', y_3'' \overset{\$}{\leftarrow} \mathbb{Z}_p$ and $\alpha, \beta \overset{\$}{\leftarrow} \mathbb{Z}_p^{\times}$, and (implicitly) sets

$$\begin{aligned}&x_0 := \frac{x_0' + y_0}{\alpha},\ x_1 := \frac{x_1' + y_1}{\alpha},\ \text{where } y_1 := y_1' + c_2 y_1'',\\ &x_2 := \frac{x_2' + c_2}{\alpha},\ y_2 := c_2,\ x_3 := \frac{x_3' + y_3}{\alpha},\ \text{where } y_3 := y_3' + c_2 y_3''.\end{aligned}$$

$\mathcal{B}$ creates

$$\begin{aligned}&z := e(g_1, g_2)^{-x_0'},\quad u_1 := g_1^{-x_1'},\quad w_1 := g_1^{-x_2'},\quad h_1 := g_1^{-x_3'},\quad \chi_1 := g_1^{-x_0'\beta},\\ &g_2^{x_1} := g_2^{\frac{x_1' + y_1'}{\alpha}} (g_2^{c_2})^{\frac{y_1''}{\alpha}},\quad g_2^{y_1} := g_2^{y_1'} (g_2^{c_2})^{y_1''},\\ &g_2^{x_2} := g_2^{\frac{x_2'}{\alpha}} (g_2^{c_2})^{\frac{1}{\alpha}},\quad g_2^{y_2} := g_2^{c_2},\quad g_2^{x_3} := g_2^{\frac{x_3' + y_3'}{\alpha}} (g_2^{c_2})^{\frac{y_3''}{\alpha}},\quad g_2^{y_3} := g_2^{y_3'} (g_2^{c_2})^{y_3''}.\end{aligned}$$

$\mathcal{B}$ sends $mpk := (g_1, g_1^{\alpha}, u_1, w_1, h_1, \chi_1, g_2, g_2^{x_1}, g_2^{x_2}, g_2^{x_3}, g_2^{y_1}, g_2^{y_2}, g_2^{y_3}, z, g_2^{\beta x_0}, g_2^{\beta y_0}, g_2^{\frac{1}{\beta}})$ to $\mathcal{A}$. Note that $\mathcal{B}$ knows a master key $msk := (g_2^{y_0}, g_2^{-x_0})$.

KeyGen oracle. Let $\mathtt{I}_i$ $(1 \leq i \leq q)$ be an i-th identity issued to the oracle. $\mathcal{B}$ creates $k - 1$ semi-functional keys, and embeds Z into the k-th keys. The rest of keys are normal.

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

Case $i = k$: $\mathcal{B}$ creates a semi-functional key by embedding Z as follows: $\mathcal{B}$ computes

$$\begin{aligned}
D_1 &:= Z,\\
D_1' &:= g_2^{y_0} (g_2^{c_1})^{\mathtt{I}_k y_1' + y_3'} Z^{\mathtt{I}_k y_1'' + y_3''},\\
D_2 &:= \Big((g_2^{c_1})^{x_2'} Z \Big)^{-\frac{1}{\alpha}},\\
D_2' &:= g_2^{-\frac{x_0'}{\alpha}} (g_2^{c_1})^{-\frac{\mathtt{I}_k (x_1' + y_1') + x_3' + y_3'}{\alpha}} g_2^{-\frac{y_0}{\alpha}} Z^{-\frac{\mathtt{I}_k y_1'' + y_3''}{\alpha}},\\
D_3 &:= g_2^{c_1}.
\end{aligned}$$

$\mathcal{B}$ sets $SK_{\mathtt{I}_k} := (D_1, D_1', D_2, D_2', D_3)$. If $b = 0$, then it is easy to see that the above keys are normal by setting $r := c_1$. If $b = 1$, then the above ciphertext is semi-functional since it holds

$$\begin{aligned}
D_1 :=& Z = g_2^{c_1 c_2 + \gamma} = g_2^{y_2 r + \gamma},\\
D_1' :=& g_2^{y_0} (g_2^{c_1})^{\mathtt{I}_k y_1' + y_3'} Z^{\mathtt{I}_k y_1'' + y_3''}\\
=& g_2^{y_0 + c_1 (\mathtt{I}_k (y_1' + c_2 y_1'') + y_3' + c_2 y_3'')} g_2^{\gamma (\mathtt{I}_k y_1'' + y_3'')} = g_2^{y_0 + r (\mathtt{I}_k y_1 + y_3)} g_2^{\gamma \phi},\\
D_2 :=& \Big((g_2^{c_1})^{x_2'} Z \Big)^{-\frac{1}{\alpha}} = g_2^{-\frac{c_1 (x_2' + c_2)}{\alpha}} g_2^{-\frac{\gamma}{\alpha}} = g_2^{-r x_2} g_2^{-\frac{\gamma}{\alpha}},\\
D_2' :=& g_2^{-\frac{x_0'}{\alpha}} (g_2^{c_1})^{-\frac{\mathtt{I}_k (x_1' + y_1') + x_3' + y_3'}{\alpha}} g_2^{-\frac{y_0}{\alpha}} Z^{-\frac{\mathtt{I}_k y_1'' + y_3''}{\alpha}}\\
=& g_2^{-\frac{(x_0' + y_0) + c_1 (\mathtt{I}_k (x_1' + y_1' + c_2 y_1'') + (x_3' + y_3' + c_2 y_3''))}{\alpha}} g_2^{-\frac{\gamma (\mathtt{I}_k y_1'' + y_3'')}{\alpha}}\\
=& g_2^{-x_0 - r (\mathtt{I}_k x_1 + x_3)} g_2^{-\frac{\gamma \phi}{\alpha}},
\end{aligned}$$

where $Z := g_2^{c_1 c_2 + \gamma}$, $r := c_1$, and $\phi := \mathtt{I}_k y_1'' + y_3''$. Since y_1'' and y_3'' are chosen uniformly at random, ϕ is also uniformly distributed.

Case $i > k$: $\mathcal{B}$ creates and returns normal keys by using the master key.

Challenge. $\mathcal{B}$ receives $(M_0^*, M_1^*, \mathtt{I}^*)$ from $\mathcal{A}$. $\mathcal{B}$ chooses $d \xleftarrow{\$} \{0, 1\}$. However, $\mathcal{B}$ cannot create a semi-functional ciphertext for $\mathtt{I}^*$ without knowledge of c_2 (and hence y_1 and y_3). To generate the semi-functional ciphertext without the knowledge, $\mathcal{B}$ sets

$$\widetilde{\mathtt{tag}}^* := - \mathtt{I}^* y_1'' - y_3''.$$

Since y_1'' and y_3'' are chosen uniformly at random, probability distribution of $\widetilde{\mathtt{tag}}^*$ is also uniformly at random from $\mathcal{A}$'s view. Then, $\mathcal{B}$ chooses $t \xleftarrow{\$} \mathbb{Z}_p$ and $\mu \xleftarrow{\$} \mathbb{Z}_p^\times$, and computes

$$\begin{aligned}
\tilde{C}_0^* :=& M_d^* z^t e(g_1, g_2)^{-x_0 \mu} = M_d^* e(g_1, g_2)^{-x_0 (\alpha t + \mu) + y_0 t},\\
\tilde{C}_1^* :=& g_1^t,\\
\tilde{C}_2^* :=& g_1^{\alpha t + \mu}\\
\tilde{C}_3^* :=& \Big(u_1^{\mathtt{I}^*} w_1^{\widetilde{\mathtt{tag}}^*} h_1 \Big)^t g_1^{-\frac{\mu}{\alpha} (\mathtt{I}^* (x_1' + y_1') + x_2' \widetilde{\mathtt{tag}}^* + x_3' + y_3')}\\
=& \Big(u_1^{\mathtt{I}^*} w_1^{\widetilde{\mathtt{tag}}^*} h_1 \Big)^t g_1^{-\frac{\mu}{\alpha} (\mathtt{I}^* (x_1' + y_1') + x_2' \widetilde{\mathtt{tag}}^* + x_3' + y_3')} g_1^{-\frac{c_2 \mu}{\alpha} (\mathtt{I}^* y_1'' + \widetilde{\mathtt{tag}}^* + y_3'')}\\
=& \Big(u_1^{\mathtt{I}^*} w_1^{\widetilde{\mathtt{tag}}^*} h_1 \Big)^t g_1^{\mu (\mathtt{I}^* x_1 + x_2 \widetilde{\mathtt{tag}}^* + x_3)}.
\end{aligned}$$

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

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

Lemma 3. $|S_q - S_{\mathsf{Final}}| \leq 2Adv_{\mathcal{G},\mathcal{B}}^{ADDH1}(\lambda)$.

Proof. At the beginning, a PPT adversary $\mathcal{B}$ receives an instance $(g_1, g_1^{dc_3}, g_1^{c_1}, g_1^{c_2}, g_2, g_2^d, g_2^{c_2c_3}, g_2^{dc_3}, g_2^{\frac{1}{c_3}}, Z)$ of the ADDH1 problem. Then, $\mathcal{B}$ randomly chooses $x_1, x_2, x_3, y_1', y_2', y_3' \xleftarrow{\$} \mathbb{Z}_p$ and $\alpha \xleftarrow{\$} \mathbb{Z}_p^\times$, and (implicitly) sets

$$x_0 := c_2, \quad y_0' := d, \quad y_0 := x_0\alpha + y_0', \quad y_1 := x_1\alpha + y_1', \quad y_2 := x_2\alpha + y_2',$$
$$y_3 := x_3\alpha + y_3', \quad \beta := c_3, \quad \beta x_0 := c_2c_3, \quad \beta y_0 := \beta(x_0\alpha + y_0') = \alpha c_2c_3 + dc_3.$$

Then, $\mathcal{B}$ creates

$$z := e(g_1, g_2^d) = e(g_1, g_2)^{y_0'}, \quad u_1 := g_1^{y_1'}, \quad w_1 := g_1^{y_2'}, \quad h_1 := g_1^{y_3'}, \quad \chi_1 := g_1^{dc_3} = g_1^{\beta y_0'},$$
$$g_2^{\beta x_0} := g_2^{c_2c_3}, \quad g_2^{\beta y_0} := (g_2^{c_2c_3})^\alpha g_2^{dc_3}, \quad g_2^{\frac{1}{\beta}} := g_2^{\frac{1}{c_3}}.$$

$\mathcal{B}$ sends $mpk := (g_1, g_1^\alpha, u_1, w_1, h_1, \chi_1, g_2, g_2^{x_1}, g_2^{x_2}, g_2^{x_3}, g_2^{y_1}, g_2^{y_2}, g_2^{y_3}, z, g_2^{\beta x_0}, g_2^{\beta y_0}, g_2^{\frac{1}{\beta}})$ to $\mathcal{A}$. Note that $\mathcal{B}$ does not know a master key $msk := (g_2^{y_0}, g_2^{-x_0})$.

KeyGen oracle. When receiving a query $\mathtt{I}$, $\mathcal{B}$ chooses $r, \phi' \xleftarrow{\$} \mathbb{Z}_p$ and $\gamma \xleftarrow{\$} \mathbb{Z}_p^\times$, and (implicitly) sets

$$\phi' := x_0 + (x_1\mathtt{I} + x_3)r + \frac{\gamma\phi}{\alpha}.$$

Then $\mathcal{B}$ computes

$$D_1 := g_2^{y_2r+\gamma},$$
$$D_1' := g_2^d g_2^{(y_1'\mathtt{I}+y_3')r+\alpha\phi'} = g_2^{x_0\alpha+y_0'+((x_1\alpha+y_1')\mathtt{I}+x_3\alpha+y_3')r+\gamma\phi} = g_2^{y_0+(y_1\mathtt{I}+y_3)r+\gamma\phi},$$
$$D_2 := g_2^{-x_2r-\frac{\gamma}{\alpha}},$$
$$D_2' := g_2^{-\phi'} = g_2^{-x_0-(x_1\mathtt{I}+x_3)r-\frac{\gamma\phi}{\alpha}},$$
$$D_3 := g_2^r.$$

$\mathcal{B}$ sends $SK_{\mathtt{I}} := (D_1, D_1', D_2, D_2', D_3)$ to $\mathcal{A}$.

Challenge. $\mathcal{B}$ receives $(M_0^*, M_1^*, \mathtt{I}^*)$ from $\mathcal{A}$. $\mathcal{B}$ chooses $d \xleftarrow{\$} \{0,1\}$. $\mathcal{B}$ chooses $t, \texttt{tag}^* \xleftarrow{\$} \mathbb{Z}_p$ and computes

$$C_0^* := M_d^* \cdot e(g_1, g_2^d)^t e(Z, g_2)^{-1}, \ C_1^* := g_1^t, \ C_2^* := g_1^{\alpha t} g_1^{c_1},$$
$$C_3^* := (u_1^{\mathtt{I}^*} w_1^{\texttt{tag}^*} h_1)^t (g_1^{c_1})^{-x_1\mathtt{I}^*-x_2\texttt{tag}^*-x_3}.$$

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

If $b = 0$, then the above ciphertext is semi-functional one of M_d^* by setting $\mu := c_1$. If $b = 1$, then the above ciphertext is semi-functional one of a random element of $\mathbb{G}_T$ since it holds

$$C_0^* = M_d^* \cdot e(g_1, g_2)^{y_0't-x_0\mu-\eta}$$

$$
\begin{aligned}
&= M_d^* \cdot e(g_1, g_2)^{-x_0\alpha t + y_0 t - x_0\mu - \eta} \\
&= M_d^* \cdot e(g_1, g_2)^{-x_0(\alpha t + \mu) + y_0 t} e(g_1, g_2)^{-\eta} \\
&= R \cdot e(g_1, g_2)^{-x_0(\alpha t + \mu) + y_0 t},
\end{aligned}
$$

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

After receiving d' from $\mathcal{A}$, $\mathcal{B}$ sends $b' = 1$ to the challenger of the ADDH1 problem if $d' = d$. Otherwise, $\mathcal{B}$ sends $b' = 0$ to the challenger. □

Proof of Theorem 2. From Lemmas 1–3, we have $Adv^{\mathit{IND\text{-}ID\text{-}CPA}}_{\Pi_{\text{JR}},\mathcal{A}}(\lambda) \leq 2Adv^{\mathit{DDH1}}_{\mathcal{G},\mathcal{B}}(\lambda) + 2q \cdot Adv^{\mathit{DDH2}}_{\mathcal{G},\mathcal{B}}(\lambda) + 2Adv^{\mathit{ADDH1}}_{\mathcal{G},\mathcal{B}}(\lambda) \leq 4Adv^{\mathit{ADDH1}}_{\mathcal{G},\mathcal{B}}(\lambda) + 2q \cdot Adv^{\mathit{DDH2}}_{\mathcal{G},\mathcal{B}}(\lambda)$. □

4 Our Construction

We construct an RIBE scheme based on the original Jutla-Roy IBE, and prove that the security of the proposed scheme relies on that of the modified Jutla-Roy IBE. An RIBE scheme $\Pi =$(Setup, SKGen, KeyUp, DKGen, Enc, Dec, Revoke) is constructed as follows.

- $\mathsf{Setup}(\lambda, N)$: It runs $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, g_1, g_2, e) \leftarrow \mathcal{G}$. It chooses $x_0, y_0, x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4, x_5, y_5 \xleftarrow{\$} \mathbb{Z}_p$ and $\alpha \xleftarrow{\$} \mathbb{Z}_p^\times$, and sets

$$
\begin{aligned}
&z = e(g_1, g_2)^{-x_0\alpha + y_0},\ u_1 := g_1^{-x_1\alpha + y_1},\ w_1 := g_1^{-x_2\alpha + y_2}, \\
&h_1 := g_1^{-x_3\alpha + y_3},\ v_1 := g_1^{-x_4\alpha + y_4},\ \hat{v}_1 := g_1^{-x_5\alpha + y_5}.
\end{aligned}
$$

  Let $\mathtt{BT}$ be a binary tree that has N leaves, where N is a power of two for simplicity. It outputs

$$
\begin{aligned}
&mpk := (g_1, g_1^\alpha, u_1, w_1, h_1, v_1, \hat{v}_1, g_2, g_2^{x_1}, g_2^{x_2}, \ldots, g_2^{x_5}, g_2^{y_1}, g_2^{y_2}, \ldots, g_2^{y_5}, z), \\
&msk := (g_2^{y_0}, g_2^{-x_0}),
\end{aligned}
$$

  $st := \mathtt{BT}$, and $RL := \emptyset$.

- $\mathsf{SKGen}(st, \mathtt{I})$: Parse st as $\mathtt{BT}$. It randomly chooses an unassigned leaf η from $\mathtt{BT}$, and stores $\mathtt{I}$ in the node η. For each node $\theta \in \mathsf{Path}(\mathtt{BT}, \eta)$, it recalls P_θ if it was defined. Otherwise, it chooses $P_\theta \xleftarrow{\$} \mathbb{G}_2$ and stores P_θ in the node θ. Then, it chooses $r_\theta \xleftarrow{\$} \mathbb{Z}_p$ and it computes

$$
\begin{aligned}
&\mathtt{SK}_{1,\theta} := (g_2^{y_2})^{r_\theta},\ \mathtt{SK}'_{1,\theta} := P_\theta\Big((g_2^{y_1})^{\mathtt{I}} g_2^{y_3}\Big)^{r_\theta}, \\
&\mathtt{SK}_{2,\theta} := (g_2^{x_2})^{-r_\theta},\ \mathtt{SK}'_{2,\theta} := P_\theta\Big((g_2^{x_1})^{\mathtt{I}} g_2^{x_3}\Big)^{-r_\theta},\ \mathtt{SK}_{3,\theta} := g_2^{r_\theta}.
\end{aligned}
$$

  It outputs $sk_{\mathtt{I}} := \{(\mathtt{SK}_{1,\theta}, \mathtt{SK}'_{1,\theta}, \mathtt{SK}_{2,\theta}, \mathtt{SK}'_{2,\theta}, \mathtt{SK}_{3,\theta})\}_{\theta \in \mathsf{Path}(\mathtt{BT},\eta)}$.

- $\mathsf{KeyUp}(msk, st, RL, T)$: Parse msk as $(\mathtt{MK}_1, \mathtt{MK}_2)$. For each node $\theta \in \mathsf{KUNode}(\mathtt{BT}, RL, T)$, it recalls P_θ if it was defined. Otherwise, it chooses $P_\theta \xleftarrow{\$} \mathbb{G}_2$ and stores P_θ in the node θ. It chooses $s_\theta \xleftarrow{\$} \mathbb{Z}_p$ and computes

$$
\mathtt{KU}'_{1,\theta} := P_\theta^{-1}\mathtt{MK}_1\Big((g_2^{y_4})^T g_2^{y_5}\Big)^{s_\theta},\ \mathtt{KU}'_{2,\theta} := P_\theta^{-1}\mathtt{MK}_2\Big((g_2^{x_4})^T g_2^{x_5}\Big)^{-s_\theta},\ \mathtt{KU}_{3,\theta} := g_2^{s_\theta}.
$$

  It outputs $ku_T := \{(\mathtt{KU}'_{1,\theta}, \mathtt{KU}'_{2,\theta}, \mathtt{KU}_{3,\theta})\}_{\theta \in \mathsf{KUNode}(\mathtt{BT},RL,T)}$.

- $\mathsf{DKGen}(sk_{\mathtt{I}}, ku_T)$: Parse $sk_{\mathtt{I}}$ and ku_T as $\{(\mathtt{SK}_{1,\theta}, \mathtt{SK}'_{1,\theta}, \mathtt{SK}_{2,\theta}, \mathtt{SK}'_{2,\theta}, \mathtt{SK}_{3,\theta})\}_{\theta\in\Theta_{\mathrm{SK}}}$ and $\{(\mathtt{KU}'_{1,\theta}, \mathtt{KU}'_{2,\theta}, \mathtt{KU}_{3,\theta})\}_{\theta\in\Theta_{\mathrm{KU}}}$, respectively. It outputs $\perp$ if $\Theta_{\mathrm{SK}} \cap \Theta_{\mathrm{KU}} = \emptyset$. Otherwise, for some $\theta \in \Theta_{\mathrm{SK}} \cap \Theta_{\mathrm{KU}}$, it computes as follows. It chooses $R, S \xleftarrow{\$} \mathbb{Z}_p$ and computes

$$\mathtt{DK}_1 := \mathtt{SK}_{1,\theta}(g_2^{y_2})^R,\ \ \mathtt{DK}'_1 := \mathtt{SK}'_{1,\theta}\mathtt{KU}'_{1,\theta}\Big((g_2^{y_1})^{\mathtt{I}}g_2^{y_3}\Big)^R\Big((g_2^{y_4})^T g_2^{y_5}\Big)^S,$$
$$\mathtt{DK}_2 := \mathtt{SK}_{2,\theta}(g_2^{x_2})^{-R},\ \ \mathtt{DK}'_2 := \mathtt{SK}'_{2,\theta}\mathtt{KU}'_{2,\theta}\Big((g_2^{x_1})^{\mathtt{I}}g_2^{x_3}\Big)^{-R}\Big((g_2^{x_4})^T g_2^{x_5}\Big)^{-S},$$
$$\mathtt{DK}_3 := \mathtt{SK}_{3,\theta}g_2^R,\ \ \mathtt{DK}_4 := \mathtt{KU}_{3,\theta}g_2^S.$$

  It outputs $dk_{\mathtt{I},T} := (\mathtt{DK}_1, \mathtt{DK}'_1, \mathtt{DK}_2, \mathtt{DK}'_2, \mathtt{DK}_3, \mathtt{DK}_4)$.

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

$$C_0 := Mz^t,\ C_1 := g_1^t,\ C_2 := (g_1^{\alpha})^t,\ C_3 := \Big(u_1^{\mathtt{I}}w_1^{\mathtt{tag}}h_1\Big)^t,\ C_4 := (v_1^T\hat{v}_1)^t.$$

  It outputs $C_{\mathtt{I},T} := (C_0, C_1, C_2, C_3, C_4, \mathtt{tag})$.

- $\mathsf{Dec}(dk_{\mathtt{I},T}, C_{\mathtt{I},T})$: Parse $dk_{\mathtt{I},T}$ and $C_{\mathtt{I},T}$ as $(\mathtt{DK}_1, \mathtt{DK}'_1, \mathtt{DK}_2, \mathtt{DK}'_2, \mathtt{DK}_3, \mathtt{DK}_4)$ and $(C_0, C_1, C_2, C_3, C_4, \mathtt{tag})$, respectively. It computes

$$M = \frac{C_0 e(C_3, \mathtt{DK}_3)e(C_4, \mathtt{DK}_4)}{e(C_1, \mathtt{DK}_1^{\mathtt{tag}}\mathtt{DK}'_1)e(C_2, \mathtt{DK}_2^{\mathtt{tag}}\mathtt{DK}'_2)}.$$

- $\mathsf{Revoke}(\mathtt{I}, T, RL, st)$: Output $RL := RL \cup \{(\mathtt{I}, T)\}$.

We show the correctness of our RIBE scheme Π.

First, we show the correctness of the DKGen algorithm. Parse $sk_{\mathtt{I}}$ and ku_T as $\{sk_\theta\}_{\theta\in\Theta_{\mathrm{SK}}} = \{(\mathtt{SK}_{1,\theta}, \mathtt{SK}'_{1,\theta},\ \mathtt{SK}_{2,\theta}, \mathtt{SK}'_{2,\theta}, \mathtt{SK}_{3,\theta})\}_{\theta\in\Theta_{\mathrm{SK}}}$ and $\{ku_\theta\}_{\theta\in\Theta_{\mathrm{KU}}} = \{(\mathtt{KU}'_{1,\theta},\ \mathtt{KU}'_{2,\theta}, \mathtt{KU}_{3,\theta})\}_{\theta\in\Theta_{\mathrm{KU}}}$, respectively. Suppose that r_θ and s_θ denotes internal randomnesses of $sk_{\mathtt{I}}$ and ku_T, respectively. Then, for any $\theta \in \Theta_{\mathrm{SK}} \cap \Theta_{\mathrm{KU}}$, we have

$$\begin{aligned}
\mathtt{DK}_1 &:= \mathtt{SK}_{1,\theta}(g_2^{y_2})^R = (g_2^{y_2})^{R+r_\theta} = (g_2^{y_2})^{\hat{R}},\\
\mathtt{DK}'_1 &:= \mathtt{SK}'_{1,\theta}\mathtt{KU}'_{1,\theta}\Big((g_2^{y_1})^{\mathtt{I}}g_2^{y_3}\Big)^R\Big((g_2^{y_4})^T g_2^{y_5}\Big)^S\\
&= g_2^{y_0}\Big((g_2^{y_1})^{\mathtt{I}}g_2^{y_3}\Big)^{R+r_\theta}\Big((g_2^{y_4})^T g_2^{y_5}\Big)^{S+s_\theta} = g_2^{y_0}\Big((g_2^{y_1})^{\mathtt{I}}g_2^{y_3}\Big)^{\hat{R}}\Big((g_2^{y_4})^T g_2^{y_5}\Big)^{\hat{S}},\\
\mathtt{DK}_2 &:= \mathtt{SK}_{2,\theta}(g_2^{x_2})^{-R} = (g_2^{x_2})^{-(R+r_\theta)} = (g_2^{x_2})^{-\hat{R}},\\
\mathtt{DK}'_2 &:= \mathtt{SK}'_{2,\theta}\mathtt{KU}'_{2,\theta}\Big((g_2^{x_1})^{\mathtt{I}}g_2^{x_3}\Big)^{-R}\Big((g_2^{x_4})^T g_2^{x_5}\Big)^{-S},\\
&= g_2^{-x_0}\Big((g_2^{x_1})^{\mathtt{I}}g_2^{x_3}\Big)^{-(R+r_\theta)}\Big((g_2^{x_4})^T g_2^{x_5}\Big)^{-(S+s_\theta)}\\
&= g_2^{-x_0}\Big((g_2^{x_1})^{\mathtt{I}}g_2^{x_3}\Big)^{-\hat{R}}\Big((g_2^{x_4})^T g_2^{x_5}\Big)^{-\hat{S}},\\
\mathtt{DK}_3 &:= \mathtt{SK}_{3,\theta}g_2^R = g_2^{R+r_\theta} = g_2^{\hat{R}},\\
\mathtt{DK}_4 &:= \mathtt{KU}_{3,\theta}g_2^S = g_2^{S+s_\theta} = g_2^{\hat{S}},
\end{aligned}$$

where $R, S \xleftarrow{\$} \mathbb{Z}_p$, $\hat{R} := R + r_\theta$, and $\hat{S} := S + s_\theta$.

We then show the decryption correctness. Suppose that $dk_{\mathtt{I},T}$ is correctly generated as above. Parse $dk_{\mathtt{I},T}$ and $C_{\mathtt{I},T}$ as $(\mathtt{DK}_1, \mathtt{DK}'_1, \mathtt{DK}_2, \mathtt{DK}'_2, \mathtt{DK}_3, \mathtt{DK}_4)$ and $(C_0, C_1, C_2, C_3, C_4, \mathtt{tag})$, respectively. Then, we have

$$\begin{aligned}
&\frac{C_0 e(C_3, \mathtt{DK}_3) e(C_4, \mathtt{DK}_4)}{e(C_1, \mathtt{DK}_1^{\mathtt{tag}}\mathtt{DK}'_1) e(C_2, \mathtt{DK}_2^{\mathtt{tag}}\mathtt{DK}'_2)} \\
&\qquad = \frac{Me(g_1, g_2)^{(-x_0\alpha+y_0)t} e(g_1^{t(\mathtt{I}(-x_1\alpha+y_1)+\mathtt{tag}(-x_2\alpha+y_2)-x_3\alpha+y_3)}, g_2^{\hat{R}})}{e(g_1^t, g_2^{y_0+y_2\hat{R}\mathtt{tag}+y_0+\hat{R}(\mathtt{I}y_1+y_3)+\hat{S}(Ty_4+y_5)})} \\
&\qquad\qquad \cdot \frac{e(g_1^{t(T(-x_4\alpha+y_4)-x_5\alpha+y_5)}, g_2^{\hat{S}})}{e(g_1^{\alpha t}, g_2^{-x_0-x_2\hat{R}\mathtt{tag}-x_0-\hat{R}(\mathtt{I}x_1+x_3)-\hat{S}(Tx_4+x_5)})} \\
&\qquad = \frac{Me(g_1, g_2)^{(-x_0\alpha+y_0)t}}{e(g_1^t, g_2^{y_0}) e(g_1^{\alpha t}, g_2^{-x_0})} = M.
\end{aligned}$$

The security of the above construction is given as follows.

Theorem 3. *If the ADDH1 and DDH2 assumptions holds, then the resulting RIBE scheme* Π *is IND-RID-CPA secure.*

We show the following lemma, and we obtain Theorem 3 as a corollary of the lemma.

Lemma 4. *The proposed RIBE scheme* Π *is IND-RID-CPA secure as long as the modified Jutla-Roy IBE* Π_{JR}*, which is described in Section 3.1, is IND-ID-CPA secure.*

Proof. We construct a PPT algorithm $\mathcal{B}$ which breaks the IND-ID-CPA security of the modified Jutla-Roy IBE Π_{JR} using a PPT adversary $\mathcal{A}$ which breaks the IND-RID-CPA security of Π.

At the beginning, $\mathcal{B}$ receives a public parameter $PP = (g_1, g_1^{\alpha}, u_1, w_1, h_1, \chi_1, g_2, g_2^{x_1}, g_2^{y_1}, g_2^{x_2}, g_2^{x_3}, g_2^{y_3}, z, g_2^{x_0\beta}, g_2^{y_0\beta}, g_2^{\frac{1}{\beta}})$. $\mathcal{B}$ guesses what time period T^* will be submitted from $\mathcal{A}$ in the challenge phase, and it holds with probability $1/|\mathcal{T}|$. Once $\mathcal{B}$ finds the guess wrong, it terminates the simulation and outputs a random bit b'. We assume $\mathcal{B}$'s guess is right in the rest of the proof. $\mathcal{B}$ creates $\mathtt{BT}$ with N leaves. $\mathcal{B}$ chooses $\tilde{x}, \hat{x}, \tilde{y}, \hat{y} \xleftarrow{\$} \mathbb{Z}_p$ and (implicitly) sets

$$\begin{aligned}
&x_4 = \beta x_0 + \tilde{x}, \quad x_5 = -T^*\beta x_0 + \hat{x}, \\
&y_4 = \beta y_0 + \tilde{y}, \quad y_5 = -T^*\beta y_0 + \hat{y}, \\
&-x_4\alpha + y_4 := -(\beta x_0 + \tilde{x})\alpha + \beta y_0 + \tilde{y} = \beta(-x_0\alpha + y_0) - \alpha\tilde{x} + \tilde{y}, \\
&-x_5\alpha + y_5 := -(-T^*\beta x_0 + \hat{x})\alpha - T^*\beta y_0 + \hat{y} = -T^*\beta(-x_0\alpha + y_0) - \alpha\hat{x} + \hat{y}.
\end{aligned}$$

Then, $\mathcal{B}$ computes

$$\begin{aligned}
&g_2^{x_4} := g_2^{\beta x_0} g_2^{\tilde{x}}, \quad g_2^{x_5} := (g_2^{\beta x_0})^{-T^*} g_2^{\hat{x}}, \quad g_2^{y_4} := g_2^{\beta y_0} g_2^{\tilde{y}}, \quad g_2^{y_5} := (g_2^{\beta y_0})^{-T^*} g_2^{\hat{y}}, \\
&v_1 := g_1^{-x_4\alpha+y_4} = \chi_1 (g_1^{\alpha})^{-\tilde{x}} g_1^{\tilde{y}}, \quad \hat{v}_1 := g_1^{-x_5\alpha+y_5} = \chi_1^{-T^*} (g_1^{\alpha})^{-\hat{x}} g_1^{\hat{y}}.
\end{aligned}$$

$\mathcal{B}$ sends $mpk := (g_1, g_1^{\alpha}, u_1, w_1, h_1, v_1, \hat{v}_1, g_2, g_2^{x_1}, g_2^{x_2}, \ldots, g_2^{x_5}, g_2^{y_1}, g_2^{y_2}, \ldots, g_2^{y_5}, z)$ to $\mathcal{A}$.

$\mathcal{B}$ guesses whether an adversary $\mathcal{A}$ will issue the target identity $\mathtt{I}^*$ to the *SKGen* oracle, and when it will issue $\mathtt{I}^*$ to the (*SKGen* and) *DKGen* oracle. More precisely, let q_1 be the maximum number of identities issued to the *SKGen* and *DKGen* oracles *before the challenge phase.* $\mathcal{B}$ randomly guesses $(k^*, i^*) \in \{1, 2\} \times \{1, 2, \ldots, q_1, q_1 + 1\}$. $k^* = 1$ denotes that $\mathcal{A}$ issues a query $\mathtt{I}^*$ for $sk_{\mathtt{I}^*}$. Note that when $k^* = 1$, $\mathtt{I}^*$ is revoked before the target time period T^*. $k^* = 2$ denotes that $\mathcal{A}$

never issues a query $\mathtt{I}^*$ for $sk_{\mathtt{I}^*}$ during the game. $i^* \in \{1, 2, \dots, q_1\}$ denotes that $\mathcal{A}$ first issues $\mathtt{I}^*$ to $\mathcal{B}$ at the i^*-th identity in their queries (before the challenge phase). $i^* = q_1 + 1$ denotes that $\mathcal{A}$ issues a query $\mathtt{I}^*$ for $sk_{\mathtt{I}^*}$ after the challenge phase. In the following, for convenience we call a type-k^* adversary as in [SE13b]. Furthermore, we classify these adversarial types more specifically according to the value of i^*: $\mathcal{A}$ is said to be a type-k^*-a adversary if $i^* \in \{1, 2, \dots, q_1\}$; and a type-$k^*$-b adversary if $i^* = q_1 + 1$. Once $\mathcal{B}$ finds the guess wrong, it terminates the simulation and outputs a random bit b'. In the rest of the proof, we assume $\mathcal{B}$'s guess is right. It holds with probability $1/2(q_1 + 1)$.

Type-1-a and Type-1-b adversary. The difference of simulations between the type-1-a and type-1-b adversaries is just a way of simulating the *SKGen* and *DKGen* oracles. When $\mathcal{A}$ is the type-1-a or type-1-b adversaries, $\mathcal{B}$ simulates the oracles as follows. $\mathcal{B}$ first chooses a node η^* for a target identity $\mathtt{I}^*$ of $\mathtt{BT}$ uniformly at random in advance.

SKGen and *DKGen* oracles for the type-1-a adversary. Suppose that $\mathcal{B}$ receives a j-th identity $\mathtt{I}$ as a secret key query $\mathtt{I}$ or a decryption key query $(\mathtt{I}, T)$ from $\mathcal{A}$. $\mathcal{B}$ then returns a secret key $sk_{\mathtt{I}}$ or a decryption key $dk_{\mathtt{I},T}$ as follows.

Case $j < i^*$: $\mathcal{B}$ first transfers $\mathtt{I}$ to the *KeyGen* oracle of the IND-ID-CPA game of Π, and gets $SK_{\mathtt{I}} := (D_1, D_1', D_2, D_2', D_3)$, if $\mathcal{B}$ does not have it. $\mathcal{B}$ randomly chooses an unassigned leaf η ($\neq \eta^*$) from $\mathtt{BT}$ and stores $\mathtt{I}$ in the node η if it is not done.

***SKGen* oracle:** For each node $\theta \in \mathsf{Path}(\mathtt{BT}, \eta)$, $\mathcal{B}$ recalls P_θ if it was defined. Otherwise, it chooses $P_\theta \xleftarrow{\$} \mathbb{G}_2$ and stores P_θ in the node θ. For $\theta \in \mathsf{Path}(\mathtt{BT}, \eta)$, if $\theta \notin \mathsf{Path}(\mathtt{BT}, \eta^*)$, then $\mathcal{B}$ chooses $r_\theta \xleftarrow{\$} \mathbb{Z}_p$ and computes

$$\mathsf{SK}_{1,\theta} := D_1 (g_2^{y_2})^{r_\theta},\ \mathsf{SK}'_{1,\theta} := P_\theta D_1' \Big((g_2^{y_1})^{\mathtt{I}} g_2^{y_3} \Big)^{r_\theta},$$
$$\mathsf{SK}_{2,\theta} := D_2 (g_2^{x_2})^{-r_\theta},\ \mathsf{SK}'_{2,\theta} := P_\theta D_2' \Big((g_2^{x_1})^{\mathtt{I}} g_2^{x_3} \Big)^{-r_\theta},\ \mathsf{SK}_{3,\theta} := D_3 g_2^{r_\theta}.$$

Otherwise, $\mathcal{B}$ chooses $r_\theta \xleftarrow{\$} \mathbb{Z}_p$ and computes

$$\mathsf{SK}_{1,\theta} := (g_2^{y_2})^{r_\theta},\ \mathsf{SK}'_{1,\theta} := P_\theta \Big((g_2^{y_1})^{\mathtt{I}} g_2^{y_3} \Big)^{r_\theta},$$
$$\mathsf{SK}_{2,\theta} := (g_2^{x_2})^{-r_\theta},\ \mathsf{SK}'_{2,\theta} := P_\theta \Big((g_2^{x_1})^{\mathtt{I}} g_2^{x_3} \Big)^{-r_\theta},\ \mathsf{SK}_{3,\theta} := g_2^{r_\theta}.$$

It stores and outputs $sk_{\mathtt{I}} := \{(\mathsf{SK}_{1,\theta}, \mathsf{SK}'_{1,\theta}, \mathsf{SK}_{2,\theta}, \mathsf{SK}'_{2,\theta}, \mathsf{SK}_{3,\theta})\}_{\theta \in \mathsf{Path}(\mathtt{BT},\eta)}$.

***DKGen* oracle:** $\mathcal{B}$ creates and stores $sk_{\mathtt{I}}$ as above if $\mathtt{I}$ is first issued to the *SKGen* and *DKGen* oracles (otherwise, $\mathcal{B}$ uses the stored $sk_{\mathtt{I}}$), and runs DKGen algorithm. Note that $\mathcal{A}$ had to issue T to the *KeyUp* oracle before issuing the decryption query, and hence ku_T was already generated at that time. It outputs $dk_{\mathtt{I},T}$.

Case $j = i^*$: Then, $\mathcal{B}$ regards the received identity $\mathtt{I}$ as a target identity, and creates a secret key for $\mathtt{I}^* := \mathtt{I}$ as follows. $\mathcal{B}$ first stores $\mathtt{I}^*$ in η^*.

***SKGen* oracle:** For each node $\theta \in \mathsf{Path}(\mathtt{BT}, \eta^*)$, $\mathcal{B}$ recalls P_θ if it was defined. Otherwise, it chooses $P_\theta \xleftarrow{\$} \mathbb{G}_2$ and stores P_θ in the node θ. For $\theta \in \mathsf{Path}(\mathtt{BT}, \eta^*)$, $\mathcal{B}$ chooses $r_\theta \xleftarrow{\$} \mathbb{Z}_p$ and computes

$$\mathsf{SK}_{1,\theta} := (g_2^{y_2})^{r_\theta},\ \mathsf{SK}'_{1,\theta} := P_\theta \Big((g_2^{y_1})^{\mathtt{I}} g_2^{y_3} \Big)^{r_\theta},$$

$$\mathtt{SK}_{2,\theta} := (g_2^{x_2})^{-r_\theta},\ \mathtt{SK}'_{2,\theta} := P_\theta\Big((g_2^{x_1})^{\mathtt{I}} g_2^{x_3}\Big)^{-r_\theta},\ \mathtt{SK}_{3,\theta} := g_2^{r_\theta}.$$

It outputs $sk_{\mathtt{I}^*} := \{(\mathtt{SK}_{1,\theta}, \mathtt{SK}'_{1,\theta}, \mathtt{SK}_{2,\theta}, \mathtt{SK}'_{2,\theta}, \mathtt{SK}_{3,\theta})\}_{\theta \in \mathsf{Path}(\mathtt{BT},\eta)}$.

***DKGen* oracle:** $\mathcal{B}$ creates and stores $sk_{\mathtt{I}}$ as above if $\mathtt{I}$ is first issued to the *SKGen* and *DKGen* oracles (otherwise, $\mathcal{B}$ uses the stored $sk_{\mathtt{I}}$), and runs DKGen algorithm.

Case $j > i^*$: If $\mathtt{I} \neq \mathtt{I}^*$, then $\mathcal{B}$ performs the same procedure in the case $j < i^*$. Otherwise, $\mathcal{B}$ does the same process in the case $j = i^*$.

SKGen and *DKGen* oracles for the type-1-b adversary. The type-1-b adversary $\mathcal{A}$ issues the target identity $\mathtt{I}^*$ only after the challenge phase. Therefore, $\mathcal{B}$ already knows what identity is a target one when $\mathcal{A}$ sends the secret or decryption key query for $\mathtt{I}^*$ to $\mathcal{B}$. We show how $\mathcal{B}$ returns a secret key $sk_{\mathtt{I}}$ or a decryption key $dk_{\mathtt{I},T}$ as follows.

Case $\mathtt{I} \neq \mathtt{I}^*$: $\mathcal{B}$ performs the same procedure in the case $j < i^*$ of the simulation for the type-1-a adversary.

Case $\mathtt{I} = \mathtt{I}^*$: $\mathcal{B}$ performs the same procedure in the case $j = i^*$ of the simulation for the type-1-a adversary.

The rest of the simulations is the same for the both of the type-1-a and type-1-b adversaries.

KeyUp oracle. When $\mathcal{B}$ receives a query T from $\mathcal{A}$, for each node $\theta \in \mathsf{KUNode}(\mathtt{BT}, RL, T)$, $\mathcal{B}$ recalls P_θ if it was defined. Otherwise, it chooses $P_\theta \xleftarrow{\$} \mathbb{G}_2$ and stores P_θ in the node θ. For $\theta \in \mathsf{KUNode}(\mathtt{BT}, RL, T)$, if $\theta \notin \mathsf{Path}(\mathtt{BT}, \eta^*)$, $\mathcal{B}$ then chooses $s_\theta \xleftarrow{\$} \mathbb{Z}_p$ and computes

$$\mathtt{KU}'_{1,\theta} := P_\theta^{-1}\Big((g_2^{y_4})^T g_2^{y_5}\Big)^{s_\theta},\ \ \mathtt{KU}'_{2,\theta} := P_\theta^{-1}\Big((g_2^{x_4})^T g_2^{x_5}\Big)^{-s_\theta},\ \ \mathtt{KU}_{3,\theta} := g_2^{s_\theta}.$$

Otherwise, $\mathcal{B}$ then chooses $s_\theta \xleftarrow{\$} \mathbb{Z}_p$ and computes

$$\begin{aligned}
&\mathtt{KU}'_{1,\theta} := P_\theta^{-1}((g_2^{y_4})^T g_2^{y_5})^{s_\theta} (g_2^{\frac{1}{\beta}})^{-\frac{T\tilde{y}+\hat{y}}{T-T^*}},\\
&\mathtt{KU}'_{2,\theta} := P_\theta^{-1}((g_2^{x_4})^T g_2^{x_5})^{-s_\theta} (g_2^{\frac{1}{\beta}})^{\frac{T\tilde{x}+\hat{x}}{T-T^*}},\ \ \mathtt{KU}_{3,\theta} := g_2^{s_\theta} (g_2^{\frac{1}{\beta}})^{-\frac{1}{T-T^*}}.
\end{aligned}$$

Note that the above can be always computed since there exists no θ such that $\theta \in \mathsf{Path}(\mathtt{BT}, \eta^*)$ and $\theta \in \mathsf{KUNode}(\mathtt{BT}, RL, T^*)$ since $sk_{\mathtt{I}^*}$ is already revoked before T^*. It finally outputs $ku_T := \{(\mathtt{KU}'_{1,\theta}, \mathtt{KU}'_{2,\theta}, \mathtt{KU}_{3,\theta})\}_{\theta \in \mathsf{KUNode}(\mathtt{BT},RL,T)}$.

The simulation goes well since it holds that

$$\begin{aligned}
((g_2^{y_4})^T g_2^{y_5})^{s_\theta} (g_2^{\frac{1}{\beta}})^{-\frac{T\tilde{y}+\hat{y}}{T-T^*}} &= ((g_2^{\beta y_0+\tilde{y}})^T g_2^{-T^*\beta y_0+\hat{y}})^{s_\theta} g_2^{-\frac{T\tilde{y}+\hat{y}}{(T-T^*)\beta}}\\
&= g_2^{y_0} (g_2^{(T-T^*)\beta y_0+T\tilde{y}+\hat{y}})^{s_\theta} g_2^{-\frac{T\tilde{y}+\hat{y}}{(T-T^*)\beta}} g_2^{-y_0}\\
&= g_2^{y_0} (g_2^{(T-T^*)\beta y_0+T\tilde{y}+\hat{y}})^{s_\theta} (g_2^{(T-T^*)\beta y_0+T\tilde{y}+\hat{y}})^{-\frac{1}{(T-T^*)\beta}}\\
&= g_2^{y_0} (g_2^{(T-T^*)\beta y_0+T\tilde{y}+\hat{y}})^{s_\theta-\frac{1}{(T-T^*)\beta}}\\
&= g_2^{y_0} (g_2^{(T-T^*)\beta y_0+T\tilde{y}+\hat{y}})^{s'_\theta}\\
&= g_2^{y_0} ((g_2^{y_4})^T g_2^{y_5})^{s'_\theta},
\end{aligned}$$

$$\begin{aligned}((g_2^{x_4})^T g_2^{x_5})^{-s_\theta}(g_2^{\frac{1}{\beta}})^{\frac{T\tilde{x}+\hat{x}}{T-T^*}} &= ((g_2^{\beta x_0+\tilde{x}})^T g_2^{-T^*\beta x_0+\hat{x}})^{-s_\theta} g_2^{\frac{T\tilde{x}+\hat{x}}{(T-T^*)\beta}} \\ &= g_2^{-x_0}(g_2^{(T-T^*)\beta x_0+T\tilde{x}+\hat{x}})^{-s_\theta} g_2^{\frac{T\tilde{x}+\hat{x}}{(T-T^*)\beta}} g_2^{x_0} \\ &= g_2^{-x_0}(g_2^{(T-T^*)\beta x_0+T\tilde{x}+\hat{x}})^{-s_\theta}(g_2^{(T-T^*)\beta x_0+T\tilde{x}+\hat{x}})^{\frac{1}{(T-T^*)\beta}} \\ &= g_2^{-x_0}(g_2^{(T-T^*)\beta x_0+T\tilde{x}+\hat{x}})^{-s_\theta+\frac{1}{(T-T^*)\beta}} \\ &= g_2^{-x_0}(g_2^{(T-T^*)\beta x_0+T\tilde{x}+\hat{x}})^{-s'_\theta} \\ &= g_2^{-x_0}((g_2^{x_4})^T g_2^{x_5})^{-s'_\theta}, \\ g_2^{s_\theta}(g_2^{\frac{1}{\beta}})^{-\frac{1}{T-T^*}} &= g_2^{s_\theta-\frac{1}{(T-T^*)\beta}} = g_2^{s'_\theta},\end{aligned}$$

where $s'_\theta = s_\theta - \frac{1}{(T-T^*)\beta}$.

Challenge. When $\mathcal{B}$ receives $(M_0^*, M_1^*, \mathtt{I}^*, T^*)$ from $\mathcal{A}$, then it sends $(M_0^*, M_1^*, \mathtt{I}^*)$ to the challenger in the IND-ID-CPA game of Π_{JR}. After receiving $(C_0^*, C_1^*, C_2^*, C_3^*, \mathtt{tag}^*)$ from the challenger, $\mathcal{B}$ sets $C_4^* := (C_2^*)^{-(T^*\tilde{x}+\hat{x})}(C_1^*)^{T^*\tilde{y}+\hat{y}}$. This is well-formed since $C_4^* = (v_1^{T^*}\hat{v}_1)^t = g_1^{t(-T^*\tilde{x}\alpha+T^*\tilde{y}-\hat{x}\alpha+\hat{y})} = g_1^{-t\alpha(T^*\tilde{x}+\hat{x})+t(T^*\tilde{y}+\hat{y})}$. $\mathcal{B}$ sends $(C_0^*, C_1^*, C_2^*, C_3^*, C_4^*, \mathtt{tag}^*)$ to $\mathcal{A}$.

When $\mathcal{A}$ outputs b', then $\mathcal{B}$ transfer it. We can show the distribution of all the above transcriptions between $\mathcal{A}$ and $\mathcal{B}$ is identical to the real experiment from the viewpoint of $\mathcal{A}$ as in [SE14b, Claim 1], and therefore we omit it.

Type-2-a and Type-2-b adversary. The difference of simulations between the type-2-a and type-2-b adversaries is also a way of simulating the *DKGen* oracle. Before describing the difference, we show how $\mathcal{B}$ simulates the *SKGen* and *KeyUp* oracles.

SKGen oracle. $\mathcal{B}$ first transfers $\mathtt{I}$ to the *KeyGen* oracle of the IND-ID-CPA game of Π, and gets $SK_{\mathtt{I}} := (D_1, D'_1, D_2, D'_2, D_3)$ if $\mathcal{B}$ does not have it. $\mathcal{B}$ randomly chooses an unassigned leaf η from $\mathtt{BT}$ and stores $\mathtt{I}$ in the node η if it is not done. For each node $\theta \in \mathsf{Path}(\mathtt{BT}, \eta)$, $\mathcal{B}$ recalls P_θ if it was defined. Otherwise, it chooses $P_\theta \xleftarrow{\$} \mathbb{G}_2$ and stores P_θ in the node θ. For $\theta \in \mathsf{Path}(\mathtt{BT}, \eta)$, $\mathcal{B}$ chooses $r_\theta \xleftarrow{\$} \mathbb{Z}_p$ and computes

$$\begin{aligned}&\mathtt{SK}_{1,\theta} := D_1(g_2^{y_2})^{r_\theta},\ \mathtt{SK}'_{1,\theta} := P_\theta D'_1\Big((g_2^{y_1})^{\mathtt{I}} g_2^{y_3}\Big)^{r_\theta}, \\ &\mathtt{SK}_{2,\theta} := D_2(g_2^{x_2})^{-r_\theta},\ \mathtt{SK}'_{2,\theta} := P_\theta D'_2\Big((g_2^{x_1})^{\mathtt{I}} g_2^{x_3}\Big)^{-r_\theta},\ \mathtt{SK}_{3,\theta} := D_3 g_2^{r_\theta}.\end{aligned}$$

It stores and outputs $sk_{\mathtt{I}} := \{(\mathtt{SK}_{1,\theta}, \mathtt{SK}'_{1,\theta}, \mathtt{SK}_{2,\theta}, \mathtt{SK}'_{2,\theta}, \mathtt{SK}_{3,\theta})\}_{\theta\in\mathsf{Path}(\mathtt{BT},\eta)}$.

KeyUp oracle. When $\mathcal{B}$ receives a query T from $\mathcal{A}$, for each node $\theta \in \mathsf{KUNode}(\mathtt{BT}, RL, T)$, $\mathcal{B}$ recalls P_θ if it was defined. Otherwise, it chooses $P_\theta \xleftarrow{\$} \mathbb{G}_2$ and stores P_θ in the node θ. For $\theta \in \mathsf{KUNode}(\mathtt{BT}, RL, T)$, $\mathcal{B}$ chooses $s_\theta \xleftarrow{\$} \mathbb{Z}_p$ and computes

$$\mathtt{KU}'_{1,\theta} := P_\theta^{-1}\Big((g_2^{y_4})^T g_2^{y_5}\Big)^{s_\theta},\ \ \mathtt{KU}'_{2,\theta} := P_\theta^{-1}\Big((g_2^{x_4})^T g_2^{x_5}\Big)^{-s_\theta},\ \ \mathtt{KU}_{3,\theta} := g_2^{s_\theta}.$$

It outputs $ku_T := \{(\mathtt{KU}'_{1,\theta}, \mathtt{KU}'_{2,\theta}, \mathtt{KU}_{3,\theta})\}_{\theta\in\mathsf{KUNode}(\mathtt{BT},RL,T)}$.

DKGen oracle for the type-2-a adversary. Let q_d $(\leq q_1)$ be the maximum number of identities made queries to the *DKGen* oracle before the challenge phase. Suppose that $\mathcal{B}$ receives a j-th identity $\mathtt{I}$ as the decryption key query $(\mathtt{I}, T)$ from $\mathcal{A}$. $\mathcal{B}$ then returns a decryption key $dk_{\mathtt{I},T}$ as follows.

Case $j < i^*$: $\mathcal{B}$ creates and stores $sk_{\mathtt{I}}$ as above if $\mathtt{I}$ is first queried to the *SKGen* and *DKGen* oracles (otherwise, $\mathcal{B}$ uses the stored $sk_{\mathtt{I}}$), and runs DKGen algorithm.

Case $j = i^*$: Then, $\mathcal{B}$ regards the received identity $\mathtt{I}$ as a target identity, and creates a decryption key for $\mathtt{I}^* := \mathtt{I}$ as follows. $\mathcal{B}$ chooses $r, s \xleftarrow{\$} \mathbb{Z}_p$ and computes

$$\begin{aligned}
&\mathtt{DK}_1 := (g_2^{y_2})^r, \ \ \mathtt{DK}'_{1,\theta} := \left((g_2^{y_1})^{\mathtt{I}^*} g_2^{y_3}\right)^r ((g_2^{y_4})^T g_2^{y_5})^s (g_2^{\frac{1}{\beta}})^{-\frac{T\tilde{y}+\hat{y}}{T-T^*}},\\
&\mathtt{DK}_2 := (g_2^{x_2})^{-r}, \ \ \mathtt{DK}'_2 := \left((g_2^{x_1})^{\mathtt{I}^*} g_2^{x_3}\right)^{-r} ((g_2^{x_4})^T g_2^{x_5})^{-s} (g_2^{\frac{1}{\beta}})^{\frac{T\tilde{x}+\hat{x}}{T-T^*}},\\
&\mathtt{DK}_3 := g_2^r, \ \ \mathtt{DK}_4 := g_2^s (g_2^{\frac{1}{\beta}})^{-\frac{1}{T-T^*}}.
\end{aligned}$$

Case $j > i^*$: If $\mathtt{I} \neq \mathtt{I}^*$, then $\mathcal{B}$ performs the same procedure in the case $j < i^*$. Otherwise, $\mathcal{B}$ does the same process in the case $j = i^*$.

DKGen oracle for the type-2-b adversary. The type-2-b adversary $\mathcal{A}$ issues the target identity $\mathtt{I}^*$ only after challenge phase. Therefore, $\mathcal{B}$ does not have to guess which identity issued to the oracle is a target one. We show how $\mathcal{B}$ returns a decryption key $dk_{\mathtt{I},T}$ as follows.

Case $\mathtt{I} \neq \mathtt{I}^*$: $\mathcal{B}$ performs the same procedure in the case $j < i^*$ of the simulation for the type-2-a adversary.

Case $\mathtt{I} = \mathtt{I}^*$: $\mathcal{B}$ performs the same procedure in the case $j = i^*$ of the simulation for the type-2-a adversary.

Challenge. $\mathcal{B}$ creates the challenge ciphertext as in the challenge phase for the type-1-a and type-1-b adversary.

When $\mathcal{A}$ outputs b', then $\mathcal{B}$ transfer it. We can also show the distribution of all the above transcriptions between $\mathcal{A}$ and $\mathcal{B}$ is identical to the real experiment from the viewpoint of $\mathcal{A}$ as in [SE14b, Claim 2], and therefore we omit it.

We estimate the reduction loss. Let $\mathcal{E}_1$ be an event that $\mathcal{B}$ correctly guesses the target time period, and $\mathcal{E}_2$ be an event that $\mathcal{B}$'s guess (k^*, i^*) is right, respectively. We then have

$$\begin{aligned}
Adv_{\Pi_{\text{JR}},\mathcal{B}}^{\textit{IND-ID-CPA}}(\lambda) &= \left|\Pr[b' = b] - \frac{1}{2}\right|\\
&= \left|\Pr[b' = b \wedge \mathcal{E}_1] + \Pr[b' = b \wedge \neg\mathcal{E}_1] - \frac{1}{2}\right|\\
&= \frac{1}{|\mathcal{T}|}\left|\Pr[b' = b \mid \mathcal{E}_1] - \frac{1}{2}\right|\\
&= \frac{1}{|\mathcal{T}|}\left|\Pr[b' = b \wedge \mathcal{E}_2 \mid \mathcal{E}_1] + \Pr[b' = b \wedge \neg\mathcal{E}_2 \mid \mathcal{E}_1] - \frac{1}{2}\right|\\
&= \frac{1}{2|\mathcal{T}|(q_1+1)}\left|\Pr[b' = b \mid \mathcal{E}_1 \wedge \mathcal{E}_2] - \frac{1}{2}\right|\\
&= \frac{1}{2|\mathcal{T}|(q_1+1)} Adv_{\Pi,\mathcal{A}}^{\textit{IND-RID-CPA}}(\lambda).
\end{aligned}$$

Therefore, we have $Adv_{\Pi,\mathcal{A}}^{\textit{IND-RID-CPA}}(\lambda) \leq 8|\mathcal{T}|(q_1+1)Adv_{\mathcal{G},\mathcal{B}}^{\textit{ADDH1}}(\lambda) + 2|\mathcal{T}|q(q_1+1)Adv_{\mathcal{G},\mathcal{B}}^{\textit{DDH2}}(\lambda)$, where q is the maximum number of queries issued to the *KeyGen* oracle in the IND-ID-CPA game of Π_{JR}. □

5 Extensions

5.1 CCA Security

Remark that the Ishida-Watanabe-Shikata scheme [IWS15] achieves not only adaptive security with DKER over prime-order groups but also CCA security. They proposed two schemes. The first one employs the BCHK transformation [BCHK07], and the second one is constructed via the KEM/DEM framework. We notice that still the size of public parameter depends on the length of identity. Although their second construction relies on the underlying Kiltz-Galindo identity-based KEM [KG06], we can employ their first construction to construct a CCA-secure RIBE scheme with constant-size public parameter based on our RIBE scheme.

5.2 Server-Aided RIBE

Qin et al. [QDLL15] proposed server-aided RIBE. In their scheme, almost all of the workloads on users are delegated to an untrusted server who does not have any secret value. Briefly, their server-aided RIBE scheme is explained as follows. A master secret key of the Seo-Emura RIBE scheme is divided to two values via two-out-of-two secret sharing, say α and β. A ciphertext has two blinding factors according to α and β such as $M \cdot e(g,g)^{\alpha} e(g,g)^{\beta}$. KGC computes a secret key of the Seo-Emura RIBE scheme for $\mathtt{I}$ by using the master secret α, and sends it to the server as *public* key of $\mathtt{I}$, say $\mathsf{PK}_{\mathtt{I}}$. Since $\mathsf{PK}_{\mathtt{I}}$ is generated by employing the CS method, the size of $\mathsf{PK}_{\mathtt{I}}$ is $O(r \log(N/r))$. Moreover, KGC issues a long-term secret key to a user $\mathtt{I}$ by using the master secret β, say $\mathsf{SK}_{\mathtt{I}}$. It is particular worth noting that the size of $\mathsf{SK}_{\mathtt{I}}$ is constant, and $\mathsf{SK}_{\mathtt{I}}$ is independent of time t. Moreover, the user can compute the decryption key, say $\mathsf{DK}_{\mathtt{I},t}$ which removes the β-part blinding factor of a ciphertext, from $\mathsf{SK}_{\mathtt{I}}$ and t regardless of whether he/she is revoked or not. At time t, KGC computes key update information ku_t, and broadcasts it. If a user ID is not revoked at time t, then the server can compute a (partial) decryption key of $\mathtt{I}$ from $\mathsf{PK}_{\mathtt{I}}$ and ku_t. They call this key transformation key, say $\mathsf{TK}_{\mathtt{I},t}$, which removes the α-part blinding factor $e(g,g)^{\alpha}$ of a ciphertext. The server partially decrypts a ciphertext by using $\mathsf{TK}_{\mathtt{I},t}$, and sends the result to the user $\mathtt{I}$. The user can obtain the plaintext by removing the β-part blinding factor $e(g,g)^{\beta}$ of the partially-decrypted ciphertext by using $\mathsf{DK}_{\mathtt{I},t}$.

This construction methodology can be employed to our RIBE scheme. Then, we can construct a server-aided RIBE scheme with the same advantages of our RIBE scheme, i.e., constant-size public parameter and asymmetric pairing settings.

6 Concluding Remarks

In the context of identity-based encryption schemes, it is natural to employ dual system encryption methodology. However, as aforementioned in the introduction, if we consider revocation functionality in the identity-based cryptosystem, there is a subtle obstacle in an approach using dual system encryption methodology, in particular, in prime-order groups. To circumvent this obstacle, we revisited the proof of Seo-Emura RIBE scheme [SE13b], which does not uses dual system encryption methodology, but give a reduction to the IND-CPA security of the underlying IBE scheme. We extract several important requirements for Seo-Emura approach, and then construct a new IBE scheme satisfying all such the requirements, based on Jutla-Roy IBE scheme. Then, we construct an RIBE based on the proposed modified Jutla-Roy IBE scheme. We prove the IND-RID-CPA security of the proposed scheme under mild variants of the SXDH assumption, which are static and generically secure.

Acknowledgments. We would like to thank anonymous reviewers for valuable comments. Yohei Watanabe was supported by Grant-in-Aid for JSPS Fellows Grant Number JP16J10532. Keita Emura was supported by JSPS KAKENHI Grant Number JP16K00198. Jae Hong Seo was supported by Next-Generation Information Computing Development Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (No. NRF-2016M3C4A7937115).

References

[BB04] Dan Boneh and Xavier Boyen. Efficient selective-id secure identity-based encryption without random oracles. In Christian Cachin and JanL. Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027, pages 223–238. Springer Berlin Heidelberg, 2004.

[BCHK07] D. Boneh, R. Canetti, S. Halevi, and J. Katz. Chosen ciphertext security from identity based encryption. *SIAM Journal on Computing*, 36(5):1301–1328, 2007.

[BF01] Dan Boneh and Matt Franklin. Identity-based encryption from the Weil pairing. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139, pages 213–229. Springer Berlin Heidelberg, 2001.

[BGK08] Alexandra Boldyreva, Vipul Goyal, and Virendra Kumar. Identity-based encryption with efficient revocation. In *Proceedings of the 15th ACM conference on Computer and communications security*, CCS '08, pages 417–426, New York, NY, USA, 2008. ACM.

[BN06] Paulo S. L. M. Barreto and Michael Naehrig. Pairing-friendly elliptic curves of prime order. In Bart Preneel and Stafford Tavares, editors, *Selected Areas in Cryptography*, SAC 2005, pages 319–331, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[CLL$^+$12a] Jie Chen, Hoon Wei Lim, San Ling, Le Su, and Huaxiong Wang. Anonymous and adaptively secure revocable IBE with constant size public parameters, 2012.

[CLL$^+$12b] Jie Chen, HoonWei Lim, San Ling, Huaxiong Wang, and Khoa Nguyen. Revocable identity-based encryption from lattices. In Willy Susilo, Yi Mu, and Jennifer Seberry, editors, *Information Security and Privacy*, volume 7372, pages 390–403. Springer Berlin Heidelberg, 2012. The full version is available at `http://eprint.iacr.org/2011/583`.

[CLL$^+$14] Jie Chen, Hoon Wei Lim, San Ling, Huaxiong Wang, and Hoeteck Wee. Shorter identity-based encryption via asymmetric pairings. *Designs, Codes and Cryptography*, 73(3):911–947, 2014.

[CZ15] Shantian Cheng and Juanyang Zhang. Adaptive-ID secure revocable identity-based encryption from lattices via subset difference method. In Javier Lopez and Yongdong Wu, editors, *Information Security Practice and Experience, ISPEC 2015*, pages 283–297, Cham, 2015. Springer International Publishing.

[ESY16] Keita Emura, Jae Hong Seo, and Taek-Young Youn. Semi-generic transformation of revocable hierarchical identity-based encryption and its DBDH instantiation. *IEICE Transactions*, 99-A(1):83–91, 2016.

[Gen06] Craig Gentry. Practical identity-based encryption without random oracles. In Serge Vaudenay, editor, *Advances in Cryptology – EUROCRYPT 2006*, volume 4004, pages 445–464. Springer Berlin Heidelberg, 2006.

[GPS08] Steven D. Galbraith, Kenneth G. Paterson, and Nigel P. Smart. Pairings for cryptographers. *Discrete Applied Mathematics*, 156(16):3113 – 3121, 2008.

[IWS15] Yuu Ishida, Yohei Watanabe, and Junji Shikata. Constructions of CCA-secure revocable identity-based encryption. In Ernest Foo and Douglas Stebila, editors, *Information Security and Privacy, ACISP 2015*, volume 9144 of *Lecture Notes in Computer Science*, pages 174–191. Springer International Publishing, 2015.

[JR13] Charanjit S. Jutla and Arnab Roy. Shorter quasi-adaptive NIZK proofs for linear subspaces. In Kazue Sako and Palash Sarkar, editors, *Advances in Cryptology – ASIACRYPT 2013*, volume 8269 of *Lecture Notes in Computer Science*, pages 1–20. Springer Berlin Heidelberg, 2013.

[KG06] Eike Kiltz and David Galindo. Direct chosen-ciphertext secure identity-based key encapsulation without random oracles. In LynnMargaret Batten and Reihaneh Safavi-Naini, editors, *Information Security and Privacy*, volume 4058, pages 336–347. Springer Berlin Heidelberg, 2006.

[KG09] Eike Kiltz and David Galindo. Direct chosen-ciphertext secure identity-based key encapsulation without random oracles. *Theoretical Computer Science*, 410(47–49):5093–5111, 2009.

[Lee16] Kwangsu Lee. Revocable hierarchical identity-based encryption with adaptive security. Cryptology ePrint Archive, Report 2016/749, 2016.

[Lew12] Allison B. Lewko. Tools for simulating features of composite order bilinear groups in the prime order setting. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237, pages 318–335. Springer Berlin Heidelberg, 2012.

[LLP14] Kwangsu Lee, Dong Hoon Lee, and Jong Hwan Park. Efficient revocable identity-based encryption via subset difference methods. Cryptology ePrint Archive, Report 2014/132, 2014. `http://eprint.iacr.org/`.

[LP16] Kwangsu Lee and Seunghwan Park. Revocable hierarchical identity-based encryption with shorter private keys and update keys. Cryptology ePrint Archive, Report 2016/460, 2016. `http://eprint.iacr.org/`.

[LV09] Benoît Libert and Damien Vergnaud. Adaptive-id secure revocable identity-based encryption. In Marc Fischlin, editor, *Topics in Cryptology – CT-RSA 2009*, volume 5473, pages 1–15. Springer Berlin Heidelberg, 2009.

[LW10] Allison Lewko and Brent Waters. New techniques for dual system encryption and fully secure hibe with short ciphertexts. In Daniele Micciancio, editor, *Theory of Cryptography, TCC 2010*, volume 5978 of *Lecture Notes in Computer Science*, pages 455–479. Springer Berlin Heidelberg, 2010.

[NNL01] Dalit Naor, Moni Naor, and Jeff Lotspiech. Revocation and tracing schemes for stateless receivers. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139, pages 41–62. Springer Berlin Heidelberg, 2001.

[PLL15] S. Park, K. Lee, and D. H. Lee. New constructions of revocable identity-based encryption from multilinear maps. *IEEE Transactions on Information Forensics and Security*, 10(8):1564–1577, Aug 2015.

[QDLL15] Baodong Qin, Robert H. Deng, Yingjiu Li, and Shengli Liu. Server-aided revocable identity-based encryption. In Günther Pernul, Peter Y A Ryan, and Edgar Weippl, editors, *Computer Security – ESORICS 2015*, pages 286–304, Cham, 2015. Springer International Publishing.

[RCS12a] Somindu C. Ramanna, Sanjit Chatterjee, and Palash Sarkar. Variants of Waters' dual system primitives using asymmetric pairings. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *Public Key Cryptography — PKC 2012*, volume 7293 of *Lecture Notes in Computer Science*, pages 298–315. Springer Berlin Heidelberg, 2012.

[RCS12b] Somindu C. Ramanna, Sanjit Chatterjee, and Palash Sarkar. Variants of Waters' dual system primitives using asymmetric pairings. Cryptology ePrint Archive, Report 2012/024, 2012. `http://eprint.iacr.org/`. The full version of [RCS12a].

[RLPL16] Geumsook Ryu, Kwangsu Lee, Seunghwan Park, and Dong Hoon Lee. Unbounded hierarchical identity-based encryption with efficient revocation. In Ho-won Kim and Dooho Choi, editors, *Information Security Applications*, WISA 2015, pages 122–133, Cham, 2016. Springer International Publishing.

[RS14] Somindu C. Ramanna and Palash Sarkar. Efficient (anonymous) compact HIBE from standard assumptions. In ShermanS.M. Chow, JosephK. Liu, LucasC.K. Hui, and SiuMing Yiu, editors, *Provable Security, ProvSec 2014*, volume 8782 of *Lecture Notes in Computer Science*, pages 243–258. Springer International Publishing, 2014.

[Sch80] Jacob T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *J. ACM*, 27(4):701–717, 1980.

[SE13a] Jae Hong Seo and Keita Emura. Efficient delegation of key generation and revocation functionalities in identity-based encryption. In Ed Dawson, editor, *Topics in Cryptology – CT-RSA 2013*, volume 7779 of *Lecture Notes in Computer Science*, pages 343–358. Springer Berlin Heidelberg, 2013.

[SE13b] Jae Hong Seo and Keita Emura. Revocable identity-based encryption revisited: Security model and construction. In Kaoru Kurosawa and Goichiro Hanaoka, editors, *Public-Key Cryptography – PKC 2013*, volume 7778, pages 216–234. Springer Berlin Heidelberg, 2013.

[SE14a] Jae Hong Seo and Keita Emura. Revocable hierarchical identity-based encryption. *Theoretical Computer Science*, 542:44–62, 2014.

[SE14b] Jae Hong Seo and Keita Emura. Revocable identity-based cryptosystem revisited: Security models and constructions. *IEEE Transactions on Information Forensics and Security*, 9(7):1193–1205, July 2014.

[SE14c] Jae Hong Seo and Keita Emura. Revocable identity-based encryption with rejoin functionality. *IEICE Transactions*, 97-A(8):1806–1809, 2014.

[SE15a] Jae Hong Seo and Keita Emura. Adaptive-id secure revocable hierarchical identity-based encryption. In Keisuke Tanaka and Yuji Suga, editors, *Advances in Information and Computer Security*, volume 9241 of *Lecture Notes in Computer Science*, pages 21–38. Springer International Publishing, 2015.

[SE15b] Jae Hong Seo and Keita Emura. Revocable hierarchical identity-based encryption: History-free update, security against insiders, and short ciphertexts. In Kaisa Nyberg, editor, *Topics in Cryptology – CT-RSA 2015*, volume 9048 of *Lecture Notes in Computer Science*, pages 106–123. Springer International Publishing, 2015.

[SE16] Jae Hong Seo and Keita Emura. Revocable hierarchical identity-based encryption via history-free approach. *Theoretical Computer Science*, 615:45–60, 2016.

[Sho97] P. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, 1997.

[SLLW14] Le Su, HoonWei Lim, San Ling, and Huaxiong Wang. Revocable ibe systems with almost constant-size key update. In Zhenfu Cao and Fangguo Zhang, editors, *Pairing-Based Cryptography – Pairing 2013*, volume 8365, pages 168–185. Springer International Publishing, 2014.

[Wat05] Brent Waters. Efficient identity-based encryption without random oracles. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494, pages 114–127. Springer Berlin Heidelberg, 2005.

[Wat09] Brent Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In Shai Halevi, editor, *Advances in Cryptology – CRYPTO 2009*, volume 5677, pages 619–636. Springer Berlin Heidelberg, 2009.

[Wee16] Hoeteck Wee. Déjà Q: Encore! un petit IBE. In Eyal Kushilevitz and Tal Malkin, editors, *Theory of Cryptography, TCC 2016-A, Part II*, pages 237–258, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg.

[WES17] Yohei Watanabe, Keita Emura, and Jae Hong Seo. New revocable IBE in prime-order groups: Adaptively secure, decryption key exposure resistant, and with short public parameters. In *Topics in Cryptology – CT-RSA 2017*, 2017. (To appear).

A Omitted Description

A.1 Identity-based Encryption

An IBE scheme Π consists of four-tuple algorithms (Init, KeyGen, IBEnc, IBDec) defined as follows. For simplicity, we omit a public parameter in the input of all algorithms except for the Init algorithm.

- $(PP, MK) \leftarrow \mathsf{Init}(\lambda)$: A probabilistic algorithm for setup. It takes a security parameter λ as input and outputs a public parameter PP and a master secret key MK.
- $SK_{\mathtt{I}} \leftarrow \mathsf{KeyGen}(MK, \mathtt{I})$: An algorithm for private key generation. It takes MK and an identity $\mathtt{I} \in \mathcal{I}$ as input and outputs a secret key $SK_{\mathtt{I}}$.
- $C \leftarrow \mathsf{IBEnc}(M, \mathtt{I})$: A probabilistic algorithm for encryption. It takes $M \in \mathcal{M}$ and $\mathtt{I} \in \mathcal{I}$ as input and then outputs a ciphertext C.
- M or $\perp \leftarrow \mathsf{IBDec}(SK_{\mathtt{I}}, C)$: A deterministic algorithm for decryption. It takes $SK_{\mathtt{I},T}$ and C as input and then outputs M or $\perp$.

In the above model, we assume that Π meets the following correctness property: For all security parameter $\lambda \in \mathbb{N}$, all $(PP, MK) \leftarrow \mathsf{Init}(\lambda)$, all $M \in \mathcal{M}$, all $\mathtt{I} \in \mathcal{I}$, it holds that $M = \mathsf{IBDec}(\mathsf{KeyGen}(MK, \mathtt{I}), \mathsf{IBEnc}(M, \mathtt{I}))$.

We describe the notion of indistinguishability against chosen plaintext attack (IND-ID-CPA). Let $\mathcal{A}$ be a PPT adversary, and $\mathcal{A}$'s advantage against IND-ID-CPA security is defined by

$$Adv_{\Pi,\mathcal{A}}^{\text{IND-ID-CPA}}(\lambda) := \left| \Pr\left[b' = b \;\middle|\; \begin{array}{l} (PP, MK) \leftarrow \mathsf{Init}(\lambda), \\ (M_0^*, M_1^*, \mathtt{I}^*, state) \leftarrow \mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{find}, PP), \\ b \xleftarrow{\$} \{0,1\}, \\ C^* \leftarrow \mathsf{IBEnc}(M_b^*, \mathtt{I}^*), \\ b' \leftarrow \mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{guess}, C^*, state) \end{array} \right] - \frac{1}{2} \right|.$$

A.2 Complexity Assumptions

We describe the DDH2v assumption, which was introduced in [RCS12a]. The authors proved the security of it in the generic bilinear group model. We furthermore describe the DDH1v assumption. This is analogous to the DDH2v assumption, and therefore its security can be proved in the same way as the DDH2v assumption.

The DDH2v assumption. Let $\mathcal{A}$ be a PPT adversary and we consider $\mathcal{A}$'s advantage against the DDH2v problem as follows.

$$Adv_{\mathcal{G},\mathcal{A}}^{DDH2v}(\lambda) := \left| \Pr\left[b' = b \;\middle|\; \begin{array}{l} D := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e) \leftarrow \mathcal{G}(\lambda), \\ d, c_1, c_2, c_3 \xleftarrow{\$} \mathbb{Z}_p,\; b \xleftarrow{\$} \{0,1\}, \\ \text{if } b = 0 \text{ then } Z := g_2^{c_1 c_2}, \text{ else } Z \xleftarrow{\$} \mathbb{G}_2, \\ b' \leftarrow \mathcal{A}(\lambda, D, g_1^d, g_1^{c_2 c_3}, g_1^{d c_3}, g_2^{c_1}, g_2^{c_2}, Z) \end{array} \right] - \frac{1}{2} \right|.$$

Definition 5 (DDH2v Assumption [RCS12a]). *The augmented DDH2v assumption relative to a generator $\mathcal{G}$ holds if for all PPT adversaries $\mathcal{A}$, $Adv_{\mathcal{G},\mathcal{A}}^{DDH2v}(\lambda)$ is negligible in λ.*

The DDH1v assumption. Let $\mathcal{A}$ be a PPT adversary and we consider $\mathcal{A}$'s advantage against the DDH1v problem as follows.

$$Adv_{\mathcal{G},\mathcal{A}}^{DDH1v}(\lambda) := \left| \Pr\left[b' = b \;\middle|\; \begin{array}{l} D := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e) \leftarrow \mathcal{G}(\lambda), \\ d, c_1, c_2, c_3 \xleftarrow{\$} \mathbb{Z}_p,\ b \xleftarrow{\$} \{0,1\}, \\ \text{if } b = 0 \text{ then } Z := g_1^{c_1 c_2},\ \text{else } Z \xleftarrow{\$} \mathbb{G}_1, \\ b' \leftarrow \mathcal{A}(\lambda, D, g_1^d, g_1^{c_1}, g_1^{c_2}, g_2^d, g_2^{c_2 c_3}, g_2^{dc_3}, Z) \end{array} \right] - \frac{1}{2} \right|.$$

Definition 6 (DDH1v Assumption)**.** *The augmented DDH1v assumption relative to a generator $\mathcal{G}$ holds if for all PPT adversaries $\mathcal{A}$, $Adv_{\mathcal{G},\mathcal{A}}^{DDH1v}(\lambda)$ is negligible in λ.*

B ADDH1 Problem in Generic Bilinear Groups

We show a security proof of the ADDH1 assumption in the generic bilinear group model to provide confidence in the assumption. The generic group model is introduced by Shoup [Sho97] to derive a lower bound on computational complexity of solving certain computational problems without looking into the actual groups structure used in a scheme. Let $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ be the Type-3 pairing. Elements of groups $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ are encoded into uniform random strings so that equality of group elements can be only tested by the adversary. We assume four oracles. Three of them simulate the group actions in $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$, respectively, and the fourth one simulates the bilinear map e. The encoding of group elements in $\mathbb{G}_1$ is modeled as an injective map $\sigma_1 : \mathbb{Z}_p \to \Sigma_1$, where $\Sigma_1 \subset \{0,1\}^*$. σ_1 maps all $x \in \mathbb{Z}_p$ to its string representation $\sigma(x)$ of $g_1^x \in \mathbb{G}_1$. Similarly, $\sigma_2 : \mathbb{Z}_p \to \Sigma_2$ and $\sigma_T : \mathbb{Z}_p \to \Sigma_T$ are defined, where $\Sigma_2, \Sigma_T \subset \{0,1\}^*$. An upper bound on the advantage of an adversary solving the ADDH1 problem in a generic bilinear group model is given by the following theorem.

Theorem 1. *Let $\mathcal{A}$ be an algorithm that attempts to solve the ADDH1 problem in the generic group model. Assume that σ_1, σ_2, and σ_T are random encoding functions for $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$, and $\mathcal{A}$ makes at most q queries to the oracles computing the group actions in $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$, and the bilinear map e. If $d, c_1, c_2, c_3, c_4 \xleftarrow{\$} \mathbb{Z}_p^\times$ and $b \xleftarrow{\$} \{0,1\}$ with $z_b := c_1 c_2$ and $z_{1-b} := c_4$, then given $\sigma_1(1)$, $\sigma_1(c_1)$, $\sigma_1(c_2)$, $\sigma_1(dc_3)$, $\sigma_1(z_0)$, $\sigma_1(z_1)$, $\sigma_2(1)$, $\sigma_2(d)$, $\sigma_2(c_2 c_3)$, $\sigma_2(dc_3)$, $\sigma_2(1/c_3)$ the advantage ϵ of $\mathcal{A}$ in solving the problem is bounded by*

$$\epsilon \leq \frac{3(q+11)^2}{4p}.$$

Proof. We consider an algorithm $\mathcal{B}$ that simulates the generic bilinear group for $\mathcal{A}$. Let $F_{i,j}$ be polynomials over $\mathbb{Z}_p[C_1, C_2, C_3, D, Z_0, Z_1]$ with 6 variables $C_1, C_2, C_3, D, Z_0, Z_1$, and $\sigma_{i,j}$ be arbitrary distinct strings from $\{0,1\}$. $\mathcal{B}$ maintains three lists of pairs, $L_i := \{(F_{i,j}, \sigma_{i,j}) : j = 0, 1, \ldots, \tau_i - 1\}$ ($i \in \{1, 2, T\}$) such that at each step τ of the game the relation $\tau_1 + \tau_2 + \tau_T = \tau + 11$ holds. At the beginning of the game (i.e., $\tau = 0$), the lists are initialized by setting $\tau_1 = 6$, $\tau_2 = 5$, $\tau_T = 0$, $F_{1,0} = 1$, $F_{1,1} = C_1$, $F_{1,2} = C_2$, $F_{1,3} = DC_3$, $F_{1,4} = Z_0$, $F_{1,5} = Z_1$, $F_{2,0} = 1$, $F_{2,1} = D$, $F_{2,2} = C_2 C_3$, $F_{2,3} = DC_3$, and $F_{2,4} = 1/C_3$. The corresponding strings are set to arbitrary distinct strings in $\{0,1\}^*$. We assume that $\mathcal{A}$ only queries the oracles on strings previously obtained from $\mathcal{B}$, and $\mathcal{B}$ can easily determine the index j of any given string $\sigma_{i,j}$ in the list L_i. $\mathcal{B}$ then starts the game by sending strings $\sigma_{1,0}, \sigma_{1,1}, \ldots, \sigma_{1,5}, \sigma_{2,0}, \sigma_{2,1}, \ldots, \sigma_{2,4}$, to $\mathcal{A}$. $\mathcal{B}$ simulates the oracles as follows.

Group actions in $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$: First, we consider $\mathbb{G}_1$. After receiving two strings σ_{1,j_1} and σ_{1,j_2} with a selection bit indicating multiplication or division from $\mathcal{A}$, $\mathcal{B}$ computes $F_{1,\tau_1} :=$

$F_{1,j_1} \pm F_{1,j_2}$. If there exists an index i with $0 \leq i < \tau_1$ such that $F_{1,\tau_1} = F_{1,i}$, then $\mathcal{B}$ sets $\sigma_{1,\tau_1} := \sigma_{1,i}$. Otherwise, it sets σ_{1,τ_1} to a uniform random string from $\{0,1\}^* \setminus \{\sigma_{1,0}, \sigma_{1,1}, \ldots, \sigma_{1,\tau_1 - 1}\}$. $\mathcal{B}$ then adds the pair $(F_{1,\tau_1}, \sigma_{1,\tau_1})$ to L_1, returns σ_{1,τ_1} to $\mathcal{A}$, and increments τ_1 by one. $\mathcal{B}$ gives similar simulations of group actions in $\mathbb{G}_2$ and $\mathbb{G}_T$.

Pairing: After receiving σ_{1,j_1} and σ_{2,j_2} from $\mathcal{A}$, $\mathcal{B}$ computes $F_{T,\tau_T} := F_{1,j_1} \cdot F_{2,j_2}$. If there exists an index i with $0 \leq i < \tau_T$ such that $F_{T,\tau_T} = F_{T,i}$, then $\mathcal{B}$ sets $\sigma_{T,\tau_T} := \sigma_{T,i}$. Otherwise, it sets σ_{T,τ_T} to a uniform random string from $\{0,1\}^* \setminus \{\sigma_{T,0}, \sigma_{T,1}, \ldots, \sigma_{T,\tau_T - 1}\}$. $\mathcal{B}$ then adds the pair $(F_{T,\tau_T}, \sigma_{T,\tau_T})$ to L_T, returns σ_{T,τ_T} to $\mathcal{A}$, and increments τ_T by one.

After at most q queries, $\mathcal{A}$ terminates and outputs a bit $b' \in \{0,1\}$. At this point, $\mathcal{B}$ chooses $d^*, c_1^*, c_2^*, c_3^*, c_4^* \xleftarrow{\$} \mathbb{Z}_p^\times$ and $b \xleftarrow{\$} \{0,1\}$, and sets $z_0^* := c_1^* c_2^*$ and $z_1^* := c_4^*$. $\mathcal{B}$ assigns $c_1^*, c_2^*, c_3^*, d^*, z_0^*, z_1^*$ to $C_1, C_2, C_3, D, Z_0, Z_1$. The simulation provided by $\mathcal{B}$ is perfect unless this assignment causes any of the following to hold.

1. $F_{1,j_1}(c_1^*, c_2^*, c_3^*, d^*, z_0^*, z_1^*) - F_{1,j_2}(c_1^*, c_2^*, c_3^*, d^*, z_0^*, z_1^*) = 0$ for some $j_1 \neq j_2$ and $F_{1,j_1} \neq F_{1,j_2}$.
2. $F_{2,j_1}(c_1^*, c_2^*, c_3^*, d^*, z_0^*, z_1^*) - F_{2,j_2}(c_1^*, c_2^*, c_3^*, d^*, z_0^*, z_1^*) = 0$ for some $j_1 \neq j_2$ and $F_{2,j_1} \neq F_{2,j_2}$.
3. $F_{T,j_1}(c_1^*, c_2^*, c_3^*, d^*, z_0^*, z_1^*) - F_{T,j_2}(c_1^*, c_2^*, c_3^*, d^*, z_0^*, z_1^*) = 0$ for some $j_1 \neq j_2$ and $F_{T,j_1} \neq F_{T,j_2}$.

Let F be an event that at least one of the above holds. As in the security proof of the DDH2v assumption in [RCS12b, Appendix B.1], which is the full version of [RCS12a], we use the following result by Schwartz [Sch80]: Let p be a prime number and $F(X_1, X_2, \ldots, X_k)$ be a non-zero polynomial in $\mathbb{Z}_p[X_1, X_2, \ldots, X_k]$ of degree m. Then, if $x_1, x_2, \ldots, x_k$ are chosen from $\mathbb{Z}_p$ uniformly at random, the probability that $F(x_1, x_2, \ldots, x_k) = 0$ is at most m/p.

We show that the simulation is perfect when F does not occur, and then b is information-theoretically hidden from the view point of $\mathcal{A}$. We note that all variables except for Z_b and Z_{1-b} are independent of b. Since Z_b is C_1C_2 which is a polynomial of degree 2, $\mathcal{A}$ will win it produces C_1C_2 using combinations of polynomials from L_1 and L_2. The only degree two polynomials that can be constructed are DC_1, DC_2, DC_3, C_2C_3 or a sum of these. $\mathcal{A}$ could also try to engineer degree three polynomials in L_T composed of C_1C_2. $\mathcal{A}$ can construct only $C_1C_2C_3$ from $\sigma_1(c_1)$ and $\sigma_2(c_2c_3)$, however, it does not have $\sigma_2(c_3)$, which is necessary for finding out b. Therefore, we have $\Pr[b = b' \mid \mathsf{F}] - 1/2$.

We then derive a bound on the probability that F occurs. For fixed j_1 and j_2, $F_{1,j_1} - F_{1,j_2}$ is a polynomial degree at most two and hence is zero at a random $d^*, c_1^*, c_2^*, c_3^*, z_0^*, z_1^*$ with probability at most $2/p$. Similarly, $F_{2,j_1} - F_{2,j_2}$ vanishes at a random $d^*, c_1^*, c_2^*, c_3^*, z_0^*, z_1^*$ with probability at most $2/p$ for fixed j_1 and j_2. For fixed j_1 and j_2, $F_{T,j_1} - F_{T,j_2}$ vanishes at a random $d^*, c_1^*, c_2^*, c_3^*, z_0^*, z_1^*$ with probability at most $3/p$ since degree of the polynomial is at most three. There are totally $\binom{\tau_1}{2}$, $\binom{\tau_2}{2}$, and $\binom{\tau_T}{2}$ pairs of polynomials from L_1, L_2, and L_T, respectively. We have $\tau_1 + \tau_2 + \tau_T = \tau + 11 \leq q + 11$ since there are at most q queries. Thus, we have

$$\Pr[\mathsf{F}] \leq \binom{\tau_1}{2}\frac{2}{p} + \binom{\tau_2}{2}\frac{2}{p} + \binom{\tau_T}{2}\frac{3}{p} \leq \epsilon \leq \frac{3(q+11)^2}{2p}.$$

Since $\Pr[b' = b] \leq \Pr[b = b' \mid \neg\mathsf{F}](1 - \Pr[\mathsf{F}]) - \Pr[\mathsf{F}] \leq 1/2 + \Pr[\mathsf{F}]/2$ and $\Pr[b' = b] \geq \Pr[b = b' \mid \neg\mathsf{F}](1 - \Pr[\mathsf{F}]) - \Pr[\mathsf{F}] = 1/2 - \Pr[\mathsf{F}]/2$, we have

$$\left|\Pr[b = b'] - \frac{1}{2}\right| \leq \frac{\Pr[\mathsf{F}]}{2} \leq \epsilon \leq \frac{3(q+11)^2}{4p}.$$

We completed the proof. □