Adaptively Secure Broadcast Encryption under Standard Assumptions with Better Efficiency

Kwangsu Lee* Dong Hoon Lee†

Abstract

In this paper, we present an efficient public-key broadcast encryption (PKBE) scheme with sub-linear size of public keys, private keys, and ciphertexts and prove its adaptive security under standard assumptions. Compared with the currently best scheme that provides adaptive security under standard assumptions and sub-linear size of various parameters, the ciphertext size of our scheme is 94% shorter and the encryption algorithm of our scheme is also 2.8 times faster than those of the currently best scheme. To achieve our scheme, we adapt the dual system encryption technique of Waters. However, there is a challenging problem to use this technique for the construction of PKBE with sub-linear size of ciphertexts such as a tag compression problem. To overcome this problem, we first devise a novel tag update technique for broadcast encryption. Using this technique, we build an efficient PKBE scheme in symmetric bilinear groups, and prove its adaptive security under standard assumptions. After that, we build another PKBE scheme in asymmetric bilinear groups and also prove its adaptive security under simple assumptions.

Keywords: Public-key encryption, Broadcast encryption, Adaptive security, Standard assumption, Bilinear maps.

1 Introduction

In broadcast encryption, a sender can efficiently send a ciphertext to the set of receivers S that is arbitrary chosen by the sender, and a receiver can decrypt the ciphertext if he belongs to the set S [15]. A trivial broadcast encryption system with linear size of ciphertexts can be built by using multiple instances of an encryption system. Therefore, a non-trivial broadcast encryption system should have sub-linear size of ciphertexts. Broadcast encryption is classified as *public key* or *symmetric key* depends on the type of keys, *stateful* or *stateless* depends on the need of private key update, and *fully-collusion resistant* or *t-collusion resistant* depends on the maximum number of collusion users.

Public-key broadcast encryption (PKBE) is a specific type of broadcast encryption such that anyone can create a ciphertext by using the the public key of broadcast encryption. Boneh, Gentry, and Waters [7] proposed the first stateless and fully-collusion resistant PKBE scheme by using the algebraic structure of bilinear groups. They first propose a simple PKBE scheme with linear size of public keys and constant size of ciphertexts, and then they proposed a generalized PKBE scheme with sub-linear size of public keys and ciphertexts. After the pioneering work of Boneh et al., many other PKBE schemes with different properties

*Korea University, Seoul, Korea. Email: `guspin@korea.ac.kr`.

†Korea University, Seoul, Korea. Email: `donghlee@korea.ac.kr`.

were proposed in bilinear groups [12, 13, 26]. However, these PKBE schemes were proven to be secure in the static security model under q-type assumptions where the value q depends on the number of users in the system. The static security model is a weaker security model since the adversary should commit the target set S^* before he receives the public key.

The right security model of PKBE is the adaptive security model where the adversary adaptively requests private keys for arbitrary chosen indexes and later selects a target subset at the challenge step [17]. Generally, a PKBE scheme in the static security model can be converted to a PKBE scheme in the adaptive security model if a simulator predicts the target set S^* of the adversary by simply selecting an arbitrary set S'. However, this method has a problem such that the probability of $S' = S^*$ is less than $1/2^N$ where N is the number of users in the system. To achieve the adaptive security, Gentry and Waters [17] proposed a new method that converts a semi-statically secure PKBE scheme to an adaptively secure one by using the two-key technique. In the semi-static security model, an adversary first commits an initial set S', and it outputs the target set S^* that is a subset of S' in the challenge step. The adversary of the semi-static security model has more flexibility compared to the static security model. The two-key technique is a method to use two keys in private keys and the decryption algorithm success if one of the two keys is given. However, their adaptively secure PKBE scheme is still secure under q-type assumptions since the security of their semi-static PKBE scheme is proven under q-type assumptions instead of standard assumptions.

In bilinear groups, q-type assumptions were widely used to build a short signature scheme in standard model [2, 3], an hierarchical identity based encryption (HIBE) scheme with constant size of ciphertexts [4], a PKBE scheme with constant size of ciphertexts [7], an attribute-based encryption (ABE) scheme [30]. However, the security of q-type assumptions is weaker than the standard assumptions as pointed by Cheon [10]. Therefore, constructing an efficient PKBE scheme that is adaptively secure under standard assumptions is a very challenging problem.

1.1 Previous Methods

Currently, there are two methods that can be used to construct an adaptively secure PKBE scheme under standard (or simple) assumptions. We briefly review these methods.

The first method is to use the *dual system encryption* technique of Waters [28]. In dual system encryption, a ciphertext and a private key can be normal or semi-functional. Additionally, it should be hard to distinguish the normal and semi-functional types, and the decryption process of any two pair of a ciphertext and a private key should be successful except the pair of the semi-functional ciphertext and the semi-functional private key. Waters proposed an HIBE scheme with linear size of ciphertexts using the dual system encryption technique that employs random tags in ciphertexts and private keys, and he proved its full model security under the DLIN and DBDH assumptions [28]. Lewko and Waters [20] proposed an HIBE scheme with constant size of ciphertexts using the dual system encryption technique. The dual system encryption technique was widely adapted to prove the full model security of HIBE, ABE, and predicate encryption (PE) [18, 21, 25]. For broadcast encryption, Waters presented a PKBE scheme with constant size of ciphertexts by removing the random tags and proved its adaptive security under the DLIN and DBDH assumptions [28, 29]. However, this PKBE scheme cannot be used for large number of users since the size of public keys and private keys is $O(N)$ where N is the number of total users in the system. Lewko et al. [19] proposed a public-key revocation encryption (PKRE) that is a special type of PKBE such that the encryption algorithm takes as input a revoked set R instead of a receiver set S, and proved its adaptive security under standard assumptions. However, their PKRE scheme cannot be used for large number of revoked users since the size of ciphertexts and the cost of encryption and decryption operations are proportional to the number of revoked users.

Table 1: Comparison between previous PKBE schemes and ours

Scheme	Adaptive	Assumption	PK Size	SK Size	CT Size	Decrypt Time
BGW [7]	No	q-Type	$O(N\lambda)$	$O(\lambda)$	$3k_p$	2P
BGW [7]	No	q-Type	$O(\sqrt{N}\lambda)$	$O(\lambda)$	$(\sqrt{N}+2)k_p$	2P
LSW [19]	No	q-Type	$O(\lambda)$	$O(\lambda)$	$(2r+2)k_p$	$2r$E + 3P
BW [9]	Yes	Static	$O(\sqrt{N}\lambda)$	$O(\sqrt{N}\lambda)$	$7\sqrt{N}k_f$	4P
GW [17]	Yes	q-Type	$O(\sqrt{N}\lambda)$	$O(\lambda)$	$5\sqrt{N}k_p$	2P + 4E
Waters [28]	Yes	DBDH, DLIN	$O(N\lambda)$	$O(N\lambda)$	$10k_p$	9P
LSW [19]	Yes	DBDH, DLIN	$O(\lambda)$	$O(\lambda)$	$(2r+8)k_p$	$r(2\mathrm{P}+\mathrm{E})$
GKSW [16]	Yes	D3DH, DLIN	$O(\sqrt{N}\lambda)$	$O(\sqrt{N}\lambda)$	$15\sqrt{N}k_p$	8P
Ours	Yes	DBDH, DLIN	$O(\sqrt{N}\lambda)$	$O(\sqrt{N}\lambda)$	$(1.5\sqrt{N}+9)k_p$	9P + 1E

λ = security parameter, N = the number of total users, r = the number of revoked users,

k_p = the bit size of group elements, E = exponentiation, P = pairing

The second method is to use the *augmented broadcast encryption* (AugBE) of Boneh and Waters [9]. AugBE is similar to PKBE except that the encryption algorithm takes as input the receiver set S and an additional index i that is hidden. The decryption algorithm of AugBE can decrypt ciphertexts if the index d in the private key satisfies $((d \in S) \wedge (i \leq d))$. An AugBE scheme is easily converted to an adaptively secure PKBE scheme if the message M and the index i in the ciphertext are hidden. An additional bonus of AugBE is that we have *trace and revoke* systems that provide broadcast encryption and traitor tracing from AugBE schemes. Boneh and Waters [9] proposed an AugBE scheme with sub-linear size of public keys and ciphertexts in composite order bilinear groups, and they proved its security under simple static assumptions. Garg et al. [16] converted the AugBE scheme of Boneh and Waters in composite order bilinear groups to an AugBE scheme in prime order bilinear groups, and proved its adaptive security under standard (DBDH and DLIN) assumptions.

Although AugBE schemes provide sub-linear size of public keys, private keys, and ciphertexts, the actual ciphertext size of AugBE schemes is quite large compared with that of PKBE schemes that are statically secure under q-type assumptions. Therefore, in this paper, we ask the following question:

> Can we build a better PKBE scheme that is adaptively secure under standard assumptions in terms of ciphertexts size?

1.2 Our Contributions

In this paper, we first propose an efficient PKBE scheme with sub-linear size of public keys and ciphertexts, and prove its adaptive security under standard (DBDH and DLIN) assumptions. Our PKBE scheme can be compared with the AugBE scheme of Garg et al. [16] since two schemes were adaptively secure under standard assumptions and provide similar asymptotic size of public keys, private keys, and ciphertexts. Although, two schemes have the similar ciphertext size of $O(\sqrt{N}\lambda)$ in big-O notation, there is a big difference in the constant value of the big-O notation. That is, the ciphertext size of our PKBE scheme is 94% shorter and the encryption algorithm is 2.8 times faster than those of Garg et al.'s AugBE scheme if we consider the

80-bit security level. The comparison of schemes is given in Table 1. The detailed efficiency comparison between schemes is also given in Section 6. Next, we propose another efficient PKBE scheme in asymmetric bilinear groups of prime order to reduce the size of ciphertexts and public keys, and prove its security under simple assumptions.

To construct our efficient PKBE schemes, we devised a novel *tag update* technique for broadcast encryption in dual system encryption. This technique is a variation of Waters' dual system encryption technique that uses random tags in ciphertexts and private keys. Though the technique of Waters cannot be used to construct a PKBE scheme with sub-linear size of public keys and ciphertexts since the random tags cannot be compressed in ciphertexts, our new technique enables the construction of PKBE with sub-linear size of public keys and ciphertexts. This technique may have independent interest.

1.3 Related Work

As mentioned, broadcast encryption allows a sender to select an arbitrary receiver subset S in the encryption algorithm and this concept was introduced by Fiat and Naor [15]. Naor, Naor, and Lotspiech [23] proposed fully collusion resistant symmetric key broadcast encryption schemes that use a tree structure and the subset cover framework. Dodis and Fazio [14] showed that the NNL schemes can be converted to the public key stetting by using the private key delegation property of HIBE schemes. A non-trivial PKBE scheme that does not rely on NNL schemes was proposed by Boneh, Gentry, and Waters [7]. After the PKBE scheme of Boneh et al., various PKBE schemes were proposed in [13, 26]. One disadvantage of PKBE is that the total number of users in the system is limited to the polynomial value of the security parameter. Identity based broadcast encryption (IBBE) is a new type of PKBE that allows the total number of users in the system can be exponential value of the security parameter. Delerablée [12] proposed the first IBBE scheme with constant size of ciphertexts. Gentry and Waters [17] constructed an IBBE scheme with sub-linear size of public keys and ciphertexts and proved its adaptive security.

Revocation encryption is another type of broadcast encryption that allows a sender to select a revocation set R instead of selecting a receiver set S in the encryption algorithm. Revocation encryption is suitable for group encryption environments where the revocation of users seldom occurs. Naor and Pinkas [24] proposed a public key revocation encryption (PKRE) scheme with t-collusion resistance. Lewko, Sahai, and Waters [19] constructed an identity based revocation encryption (IBRE) scheme with constant size of public keys and private keys.

Traitor tracing solves the problem of traitor in broadcast encryption, and it was introduced by Chor, Fiat, and Naor [11]. For example, a content distributor first broadcasts a ciphertext for legitimate receiver decoders. If a traitor hacks a legitimate decoder and builds a pirate decoder, then the distributor can run the tracing algorithm to extract an index of the traitor by interacting with the pirate decoder. After that, the distributor can take legal actions against the traitor. Boneh, Sahai, and Waters [8] proposed a fully collusion resistant traitor tricing scheme by introducing a new primitive called private linear broadcast encryption (PLBE). Generally, traitor tracing is used with broadcast encryption, and this system is called a trace & revoke system [24]. Boneh and Waters [9] presented a fully collusion resistant trace & revoke system by introducing a new primitive called augmented broadcast encryption (AugBE). Garg et al. [16] and Park et al. [27] obtained AugBE schemes in prime order groups from the AugBE scheme of Boneh and Waters.

2 Preliminaries

In this section, we first define PKBE and give the formal definition of its adaptive security model. Then we define the bilinear groups in prime order groups and introduce the complexity assumptions of our scheme.

2.1 Public-Key Broadcast Encryption

Public-key broadcast encryption (PKBE) is a specific type of broadcast encryption such that anyone can create a ciphertext for a receiver set S by using a public key. The following is the syntax of PKBE.

Definition 2.1 (Public-key broadcast encryption)**.** *A public-key broadcast encryption (PKBE) scheme consists of four algorithms **Setup**, **KeyGen**, **Encrypt**, and **Decrypt**, which are defined as follows:*

***Setup**$(1^\lambda, N)$. The setup algorithm takes as input a security parameter 1^λ and the number of receivers N. It outputs a public key PK and a master key MK.*

***KeyGen**(d, MK, PK). The key generation algorithm takes as input an index $d \in \{1,\ldots,N\}$ of a user, the master key MK, and the public key PK. It outputs a private key SK_d for the user d.*

***Encrypt**(S, PK). The encryption algorithm takes as input a subset $S \subseteq \{1,\ldots,N\}$ and the public key PK. It outputs a ciphertext header CH_S and an encryption key EK.*

***Decrypt**(CH_S, SK_d, PK). The decryption algorithm takes as input a ciphertext header CH_S for a subset S, a private key SK_d for an user index d, and the public key PK. If $d \in S$, then it outputs an encryption key EK.*

*The correctness property of PKBE is defined as follows: For all $S \in \{1,\ldots,N\}$, $d \in \{1,\ldots,N\}$, let $(PK, MK) \leftarrow$ **Setup**$(1^\lambda, N)$, $SK_d =$ **KeyGen**(d, MK, PK), and $(CH_S, EK) =$ **Encrypt**(S, PK).*

- *If $d \in S$, then **Decrypt**$(CH_S, SK_d, PK) = EK$.*
- *If $d \notin S$, then **Decrypt**$(CH_S, SK_d, PK) = \perp$ with all but negligible probability.*

Definition 2.2 (Security)**.** *The security property of PKBE under a chosen plaintext attack is defined in terms of the following experiment between a challenger $\mathcal{C}$ and a PPT adversary $\mathcal{A}$:*

1. ***Setup**: $\mathcal{C}$ runs the setup algorithm and keeps the master key MK, then it gives the public key PK to $\mathcal{A}$.*
2. ***Query**: $\mathcal{A}$ adaptively requests a private key for an index d. $\mathcal{C}$ creates a private key and gives it to $\mathcal{A}$.*
3. ***Challenge**: $\mathcal{A}$ submits a challenge subset S^* subject to the restriction that for any index d given out in the query stage, $d \notin S^*$. $\mathcal{C}$ runs the encryption algorithm **Encrypt**(S^*, PK) to obtain a header CH^* and an encryption key EK^*. Next, it flips a random coin $\gamma \in \{0,1\}$. If $\gamma = 0$, then it sets $EK_\gamma = EK^*$. Otherwise, it picks a random $EK_\gamma \in \mathcal{K}$. It then gives (CH^*, EK_γ) to $\mathcal{A}$.*
4. ***Guess**: $\mathcal{A}$ outputs a guess γ' of γ, and wins if $\gamma' = \gamma$.*

The advantage of $\mathcal{A}$ is defined as $\boldsymbol{Adv}_{\mathcal{A}}^{PKBE} = \left|\Pr[\gamma = \gamma'] - 1/2\right|$ where the probability is taken over all the randomness of the experiment. A PKBE scheme is adaptively secure under a chosen plaintext attack if for all PPT adversary $\mathcal{A}$, the advantage of $\mathcal{A}$ in the above experiment is negligible in the security parameter λ.

2.2 Bilinear Groups of Prime Order

Let $\mathbb{G}$ and $\mathbb{G}_T$ be multiplicative cyclic groups of prime p order. Let g be a generator of $\mathbb{G}$. The bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ has the following properties:

1. Bilinearity: $\forall u, v \in \mathbb{G}$ and $\forall a, b \in \mathbb{Z}_p$, $e(u^a, v^b) = e(u, v)^{ab}$.

2. Non-degeneracy: $\exists g$ such that $e(g, g)$ has order p, that is, $e(g, g)$ is a generator of $\mathbb{G}_T$.

We say that $\mathbb{G}, \mathbb{G}_T$ are bilinear groups if the group operations in $\mathbb{G}$ and $\mathbb{G}_T$ as well as the bilinear map e are all efficiently computable.

2.3 Complexity Assumptions

We introduce two standard assumptions in prime order bilinear groups. The DLIN assumption was introduced in [5]. The DBDH assumption was introduced in [1, 6].

Assumption 2.3 (Decisional Linear, DLIN [5])**.** *Let $(p, \mathbb{G}, \mathbb{G}_T, e)$ be a description of the bilinear group of prime order p. The DLIN assumption is that if the challenge values*

$$D = ((p, \mathbb{G}, \mathbb{G}_T, e),\ g, f, d, g^a, f^b) \text{ and } T$$

are given, no PPT algorithm $\mathcal{A}$ can distinguish $T = T_0 = d^{a+b}$ from $T = T_1 = d^c$ with more than a negligible advantage. The advantage of $\mathcal{A}$ is defined as $\boldsymbol{Adv}_{\mathcal{A}}^{DLIN}(\lambda) = \big|\Pr[\mathcal{A}(D, T_0) = 1] - \Pr[\mathcal{A}(D, T_1) = 1]\big|$ where the probability is taken over the random choices of $f, d, \in \mathbb{G}$ and $a, b, c \in \mathbb{Z}_p$.

Assumption 2.4 (Decisional Bilinear Diffie-Hellman, DBDH, [1, 6])**.** *Let $(p, \mathbb{G}, \mathbb{G}_T, e)$ be a description of the bilinear group of prime order p. The DBDH assumption is that if the challenge values*

$$D = ((p, \mathbb{G}, \mathbb{G}_T, e),\ g, g^a, g^b, g^c) \text{ and } T$$

are given, no PPT algorithm $\mathcal{A}$ can distinguish $T = T_0 = e(g, g)^{abc}$ from $T = T_1 = e(g, g)^d$ with more than a negligible advantage. The advantage of $\mathcal{A}$ is defined as $\boldsymbol{Adv}_{\mathcal{A}}^{DBDH}(\lambda) = \big|\Pr[\mathcal{A}(D, T_0) = 1] - \Pr[\mathcal{A}(D, T_1) = 1]\big|$ where the probability is taken over the random choices of $a, b, c, d \in \mathbb{Z}_p$.

3 Key Ideas

In this section, we describe the main idea of our construction. Before presenting the main idea, we describe the proof technique of dual system encryption since our construction rely on this technique.

Dual System Encryption. The security proof of dual system encryption consists of hybrid games that replace a normal ciphertext or a normal private key with a semi-functional ciphertext or a semi-functional private key one by one. In the first game, a normal ciphertext is replaced with a semi-functional one. In the next games, each normal private key of a private key extraction query with an index less than i is replaced with a semi-functional one. In the final game, the semi-functional ciphertext and the semi-functional private keys are given, and the session key is replaced with a random one. The merit of this hybrid proof strategy is that the security proof is simple since only one ciphertext or private key is changed, and the adversary's advantage of the final game is zero since there is no relationship between the semi-functional ciphertext and the semi-functional private keys.

However, there is a big problem in the security proof of dual system encryption. That is, the paradox of dual system encryption should be solved. The paradox is described as follows: We consider the games such that an adversary distinguishes whether the ith private key is normal or semi-functional. In this game, the simulator can create the semi-functional challenge ciphertext of a subset S and decrypt that ciphertext using the ith private key of an index d. If the ith private key is normal, then the decryption will succeed. Otherwise, the decryption will fail. Therefore, the simulator can easily distinguish the type of ith private key using the results of decryption without the help of the adversary.

We can solve this paradox of dual system encryption if we can set the decryption results of the normal ith private key and the semi-functional ith private key are the same value. Waters [28] used random tags in ciphertexts and private keys, and then changed the decryption logic of IBE from $(ID_c = ID_k)$ to $((ID_c = ID_k) \wedge (\mathrm{tag}_c \neq \mathrm{tag}_k))$ where tag_c and tag_k are random tags in the ciphertext of an identity ID_c and the private key of an identity ID_k respectively. To solve the paradox, the simulator uses a fixed function $f(x)$, and then it sets $\mathrm{tag}_c = f(ID_c)$ for the semi-functional ciphertext and $\mathrm{tag}_k = f(ID_k)$ for the ith private key. Thus if $ID_c = ID_k$, then the decryption always fails since $\mathrm{tag}_c = \mathrm{tag}_k$. For broadcast encryption, Waters proposed a PKBE scheme with liner size of public keys and constant size of ciphertexts without using the random tags [29].

Naive Approach. To construct a PKBE scheme with sub-linear size of public keys and ciphertexts, we may consider to use parallel instances of Waters' PKBE scheme [28] to balance the size of public keys and ciphertexts. This parallel construction technique is a standard way to construct sub-linear size of PKBE schemes [7, 26]. However, this parallel construction technique does not work with Waters' PKBE scheme. The problem is that if public key elements are shared between multiple instances of PKBE scheme for the construction of sub-linear size of PKBE, then the security proof of dual system encryption does not work since one public key element influences many users. The reason of this problem is that the dual system encryption technique for broadcast encryption use public key elements to overcome the paradox of dual system encryption instead of using the random tags.

To solve the previous problem, we may consider to combine the PKBE scheme that is derived from the HIBE scheme of Boneh et al. [4] and the dual system encryption technique of Waters [28] that uses random tags. However, this approach has a tag compression problem. To solve the tag compression problem, we may use a single tag value instead of multiple tags in ciphertexts. However, this simple method does not solve the paradox. To solve the paradox, the simulator should set tag_c in the ciphertext header of a subset S and tag_k in the private key of an index d as the same value if $d \in S$. However, the simulator can not set all tag_k of an index d where $d \in S$ as the same value because it can not predict the challenge subset S. Therefore, this simple method does not solve the paradox.

New Technique. To construct a PKBE scheme with sub-linear size of public keys and ciphertexts using dual system encryption, we devise a *tag update* technique. This technique uses a single tag value and changes the tag value in a private key into a new tag value when the private key is used in the decryption algorithm. At first, the private key of an index d contains tag_k and $\{z_i\}_{1 \leq i \neq d \leq m}$ values. If the private key is used in the decryption algorithm for a ciphertext header with a subset S where $d \in S$, then the tag is updated to $\mathrm{tag}'_k = \mathrm{tag}_k + \sum_{i \in S \setminus \{d\}} z_i$. To solve the paradox, a simulator fixes a function $f(S) = y + \sum_{i \in S} x_i$. Next, it sets $\mathrm{tag}_c = f(S)$ for the semi-function ciphertext header of a subset S, and it also sets $\mathrm{tag}_k = y + x_d$ and $z_j = x_j$ for the private key of an index d. If the index d is a member of the challenge subset S in the ciphertext header, then the updated tag value tag'_k that will be used for decryption is equal with tag_c. Therefore, the paradox of dual system encryption is solved even if it uses a single tag.

4 Main Construction

In this section, we present an efficient PKBE scheme based on prime order bilinear groups and prove its adaptive security under two standard assumptions.

4.1 Construction

Let N be the total number of users and $m = \lceil \sqrt{N} \rceil$. An index $d \in \{1, \ldots, N\}$ is represented as a position (d_x, d_y) in a $m \times m$ matrix where $d = (d_y - 1)m + d_x$ for some $1 \leq d_y \leq m$ and $1 \leq d_x \leq m$. Let S be a subset of $\{1, \ldots, N\}$, and define $S'_j = S \cap \{(j-1)m+1, \ldots, (j-1)m+m\}$ and $S_j = \{x - (j-1)m | x \in S'_j\} \subseteq \{1, \ldots, m\}$. A subset S is divided to subsets $S_1, \ldots, S_m$.

PKBE.Setup($1^\lambda, N$): This algorithm first generates the bilinear group $\mathbb{G}$ of prime order p of bit size $\Theta(\lambda)$. It chooses random elements $g, v, v_1, v_2, h_1, \ldots, h_m, u_1, \ldots, u_m, w \in \mathbb{G}$ and random exponents $a_1, a_2, b, \alpha \in \mathbb{Z}_p$. It outputs a master key $MK = \left(g^{a_1 \alpha}, g^{-\alpha}, v, v_1, v_2\right)$ and a public key as

$$PK = \Big(g, g^{a_1}, g^{a_2}, g^b, g^{a_1 b}, g^{a_2 b}, v v_1^{a_1}, v v_2^{a_2}, (v v_1^{a_1})^b, (v v_2^{a_2})^b, \\ h_1, \ldots, h_m,\ u_1, \ldots, u_m,\ w,\ \Omega = e(g^{a_1}, g^b)^\alpha \Big).$$

PKBE.KeyGen(d, MK, PK): This algorithm takes as input an index $d = (d_x, d_y)$, the master key MK, and the public key PK. It selects random exponents $r_1, r_2, r_3, r_4 \in \mathbb{Z}_p$ and random values $\mathrm{tag}_k, z_1, \ldots, z_m \in \mathbb{Z}_p$. It outputs a private key by implicitly including d as

$$SK_d = \Big(D_1 = g^{a_1 \alpha} v^{r_1 + r_2},\ D_2 = g^{-\alpha} v_1^{r_1 + r_2} g^{r_3},\ D_3 = (g^b)^{-r_3},\ D_4 = v_2^{r_1 + r_2} g^{r_4}, \\ D_5 = (g^b)^{-r_4},\ D_6 = (g^b)^{-r_2},\ D_7 = g^{-r_1}, \\ K_1 = (h_{d_y} u_{d_x})^{r_1} w^{\mathrm{tag}_k r_1},\ \left\{K_{2,i} = u_i^{r_1} w^{z_i r_1}\right\}_{1 \leq i \neq d_x \leq m},\ \mathrm{tag}_k,\ \{z_i\}_{1 \leq i \neq d_x \leq m} \Big).$$

PKBE.Encrypt(S, PK): This algorithm takes as input a receiver set $S \subseteq \mathcal{N}$ that divided to subsets $S_1, \ldots, S_m$ and the public key PK. It first chooses random exponents $s_1, s_2, t \in \mathbb{Z}_p$ and random values $\mathrm{tag}_{c,1}, \ldots, \mathrm{tag}_{c,m} \in \mathbb{Z}_p$. It outputs a ciphertext header by implicitly including S as

$$CH_S = \Big(E_1 = (g^b)^{s_1 + s_2},\ E_2 = (g^{a_1 b})^{s_1},\ E_3 = (g^{a_1})^{s_1},\ E_4 = (g^{a_2 b})^{s_2}, \\ E_5 = (g^{a_2})^{s_2},\ E_6 = (v v_1^{a_1})^{s_1} (v v_2^{a_2})^{s_2},\ E_7 = ((v v_1^{a_1})^b)^{s_1} ((v v_2^{a_2})^b)^{s_2} w^{-t}, \\ C_1 = g^t,\ \Big\{C_{2,j} = (h_j \prod_{i \in S_j} u_i)^t w^{\mathrm{tag}_{c,j} t}\Big\}_{1 \leq j \leq m},\ \{\mathrm{tag}_{c,j}\}_{1 \leq j \leq m} \Big)$$

and an encryption key $EK = \Omega^{s_2}$.

PKBE.Decrypt(CH_S, SK_d, PK): This algorithm takes as input a ciphertext header CH_S for a receiver set $S = S_1 \cup \cdots \cup S_m$ and a private key SK_d for an index $d = (d_x, d_y)$. If $d \notin S$, it outputs $\perp$. Otherwise it proceeds as follows:

1. It finds a subset S_{d_y} from the set S such that $d_x \in S_{d_y}$ and calculates $\mathrm{tag}'_k = \mathrm{tag}_k + \sum_{i \in S_{d_y} \setminus \{d_x\}} z_i$ from the private key.

2. If $\mathrm{tag}'_k \neq \mathrm{tag}_{c,d_y}$, then it outputs an encryption key as

$$EK = \prod_{i=1}^{7} e(E_i, D_i) \cdot \Big(e(C_1, K_1 \prod_{i \in S_{d_y} \setminus \{d_x\}} K_{2,i}) \cdot e(C_{2,d_y}, D_7) \Big)^{-1/(\mathrm{tag}'_k - \mathrm{tag}_{c,d_y})}.$$

Otherwise, it outputs $\perp$.

4.2 Correctness

Let $\mathrm{tag}'_k = \mathrm{tag}_k + \sum_{i \in S_{d_y} \setminus \{d_x\}} z_i$. If $\mathrm{tag}'_k \neq \mathrm{tag}_{c,d_y}$, then the correctness of the above PKBE scheme is easily verified as

$$\begin{aligned}
&\left(\prod_{i=1}^{7} e(E_i, D_i) \right) \cdot \left(\frac{e(C_1, K_1 \prod_{i \in S_{d_y} \setminus \{d_x\}} K_{2,i})}{e(C_{2,d_y}, D_7^{-1})} \right)^{-\frac{1}{\mathrm{tag}'_k - \mathrm{tag}_{c,d_y}}} \\
&= \Big(e(g^b, g^{a_1})^{s_2 \alpha} \cdot e(w^t, g^{r_1}) \Big) \cdot \left(\frac{e(g^t, w^{\mathrm{tag}'_k r_1})}{e(w^{\mathrm{tag}_{c,d_y}}, g^{r_1})} \right)^{-\frac{1}{\mathrm{tag}'_k - \mathrm{tag}_{c,d_y}}} \\
&= \Big(e(g^b, g^{a_1})^{s_2 \alpha} \cdot e(w^t, g^{r_1}) \Big) \cdot e(g^t, w^{r_1})^{-1} = e(g^{a_1}, g^b)^{\alpha s_2} = \Omega^{s_2}.
\end{aligned}$$

Note that we have $\mathrm{tag}'_k \neq \mathrm{tag}_{c,d_y}$ with $1 - 1/p$ probability since $\mathrm{tag}_k, z_1, \ldots, z_m, \mathrm{tag}_{c,d_y}$ are randomly chosen in $\mathbb{Z}_p$.

4.3 Security Analysis

Theorem 4.1. *The above PKBE scheme is adaptively secure under a chosen plaintext attack if the DLIN and DBDH assumptions hold. That is, for any PPT adversary $\mathcal{A}$, there exists a PPT algorithm $\mathcal{B}$ such that*

$$\boldsymbol{Adv}_{\mathcal{A}}^{PKBE} \leq (N+1) \cdot \boldsymbol{Adv}_{\mathcal{B}}^{DLIN} + \boldsymbol{Adv}_{\mathcal{B}}^{DBDH}$$

where N is the number of total users in the system.

Proof. To prove the security of our scheme, we use the dual system encryption technique of Waters [28]. We first define semi-functional private keys and ciphertext headers.

PKBE.KeyGenSF. This algorithm first creates a normal private key $SK'_d = (D'_1, \ldots, D'_7, K'_1, \{K'_{2,i}\}, \mathrm{tag}_k, \{z_i\})$. It chooses a random exponent $r_5 \in \mathbb{Z}_p$ and outputs a semi-functional private key as

$$\begin{aligned}
SK_d = \Big(& D_1 = D'_1 \cdot (g^{-a_1 a_2})^{r_5},\ D_2 = D'_2 \cdot (g^{a_2})^{r_5},\ D_3 = D'_3,\ D_4 = D'_4 \cdot (g^{a_1})^{r_5}, \\
& D_5 = D'_5,\ D_6 = D'_6,\ D_7 = D'_7,\ K_1 = K'_1,\ \{K_{2,i} = K'_{2,i}\},\ \mathrm{tag}_k,\ \{z_i\} \Big).
\end{aligned}$$

PKBE.EncryptSF. This algorithm first creates a normal ciphertext header $CH'_S = (E'_1, \ldots, E'_7, C'_1, \{C'_{2,j}\}, \{\mathrm{tag}_{c,j}\})$ and an encryption key EK'. It chooses a random exponent $s_3 \in \mathbb{Z}_p$ and outputs a semi-functional ciphertext header as

$$\begin{aligned}
CH_S = \Big(& E_1 = E'_1,\ E_2 = E'_2,\ E_3 = E'_3,\ E_4 = E'_4 \cdot (g^{a_2 b})^{s_3},\ E_5 = E'_5 \cdot (g^{a_2})^{s_3}, \\
& E_6 = E'_6 \cdot (v_2^{a_2})^{s_3},\ E_7 = E'_7 \cdot (v_2^{a_2 b})^{s_3},\ C_1 = C'_1,\ \{C_{2,j} = C'_{2,j}\},\ \{\mathrm{tag}_{c,j}\} \Big)
\end{aligned}$$

and an encryption key $EK = EK'$.

Note that if a semi-functional private key is used to decrypt a semi-functional ciphertext header, then the decryption algorithm will fail to produce a valid encryption key since it is multiplied by the additional term $e((g^{a_2 b})^{s_3}, (g^{a_1})^{r_5})$.

The security proof consists of the following sequence of hybrid games. The first game will be the original security game and the last one will be a game such that the adversary has no advantage. We define the games as follows:

Game $\mathbf{G}_0$ This game is the original adaptive security game in Section 2. In this game, the private keys and the challenge ciphertext header are normal.

Game $\mathbf{G}_1$ This game is almost identical to $\mathbf{G}_0$ except that the challenge ciphertext header is semi-functional.

Game $\mathbf{G}_2$ This game is the same with the $\mathbf{G}_1$ except that the private keys will be semi-functional. At this moment, the private keys and the challenge ciphertext header are all semi-functional. Suppose that an adversary makes at most q private key queries. We define a sequence of games $\mathbf{G}_{1,0}, \mathbf{G}_{1,1}, \dots, \mathbf{G}_{1,q}$ where $\mathbf{G}_{1,0} = \mathbf{G}_1$. In $\mathbf{G}_{1,i}$, for all jth private key queries such that $j > k$, a normal private key is given to the adversary. However, for all jth private key queries such that $j \leq k$, a semi-functional private key is given to the adversary. It is obvious that $\mathbf{G}_{1,q}$ is equal with $\mathbf{G}_2$.

Game $\mathbf{G}_3$ We now define the final game $\mathbf{G}_3$. This game differs from $\mathbf{G}_2$ in that the challenge encryption key EK^* for EK_0 is replaced by a random element. Note that in this game, the challenge ciphertext header and the challenge encryption key gives no information about the random coin γ. Therefore, the adversary can win this game with probability at most $1/2$.

Let $\mathbf{Adv}_{\mathcal{A}}^{G_j}$ be the advantage of $\mathcal{A}$ in $\mathbf{G}_j$. It is obvious that $\mathbf{Adv}_{\mathcal{A}}^{PKBE} = \mathbf{Adv}_{\mathcal{A}}^{G_0}, \mathbf{Adv}_{\mathcal{A}}^{G_3} = 0$. From the following three lemmas, we have that it is hard to distinguish $\mathbf{G}_{i-1}$ from $\mathbf{G}_i$ under the given assumptions. Therefore, we have

$$\begin{aligned}\mathbf{Adv}_{\mathcal{A}}^{PKBE} &= \mathbf{Adv}_{\mathcal{A}}^{G_0} + \sum_{i=1}^{2} (\mathbf{Adv}_{\mathcal{A}}^{G_i} - \mathbf{Adv}_{\mathcal{A}}^{G_i}) + \mathbf{Adv}_{\mathcal{A}}^{G_0} \leq \sum_{i=1}^{3} \left| \mathbf{Adv}_{\mathcal{A}}^{G_{i-1}} - \mathbf{Adv}_{\mathcal{A}}^{G_i} \right| \\ &\leq \mathbf{Adv}_{\mathcal{B}}^{DLIN} + N\mathbf{Adv}_{\mathcal{B}}^{DLIN} + \mathbf{Adv}_{\mathcal{B}}^{DBDH}.\end{aligned}$$

This completes our proof. □

Lemma 4.2. *If the DLIN assumption holds, then no polynomial-time adversary can distinguish between $\mathbf{G}_0$ and $\mathbf{G}_1$ with a non-negligible advantage.*

Proof. Suppose there exists an adversary $\mathcal{A}$ that distinguishes between $\mathbf{G}_0$ and $\mathbf{G}_1$ with a non-negligible advantage. A simulator $\mathcal{B}$ that solves the DLIN assumption using $\mathcal{A}$ is given: a challenge tuple $D = ((p, \mathbb{G}, \mathbb{G}_T, e), g, f, d, g^{c_1}, f^{c_2})$ and T where $T = d^{c_1+c_2}$ or $T = d^{c_1+c_2+c_3}$. Then $\mathcal{B}$ that interacts with $\mathcal{A}$ is described as follows: $\mathcal{B}$ first chooses random exponents $b, v', v'_1, v'_2, \alpha \in \mathbb{Z}_p$ and random elements $h_1, \dots, h_m, u_1, \dots, u_m, w \in \mathbb{G}$. It sets the master key as $MK = (g^{a_1 \alpha} = f^{\alpha}, g^{-\alpha}, v = g^{v'}, v_1 = g^{v'_1}, v_2 = g^{v'_2})$ and publishes the public key PK as

$$\begin{aligned}&g,\ g^{a_1} = f,\ g^{a_2} = d,\ g^b,\ g^{a_1 b} = f^b,\ g^{a_2 b} = d^b,\ vv_1^{a_1} = g^{v'} f^{v'_1},\ vv_2^{a_2} = g^{v'} d^{v'_2}, \\ &(vv_1^{a_1})^b = (g^{v'} f^{v'_1})^b,\ (vv_2^{a_2})^b = (g^{v'} d^{v'_2})^b,\ h_1, \dots, h_m,\ u_1, \dots, u_m,\ w,\ \Omega = e(f, g^b)^{\alpha}.\end{aligned}$$

$\mathcal{A}$ adaptively requests a private key query for an index d. To response this query, $\mathcal{B}$ simply runs the key generation algorithm to create a normal private key using the master key. Note that $\mathcal{B}$ can only create the normal private keys since it does not know a_1, a_2. In the challenge step, $\mathcal{A}$ submits a challenge set $S^* = S_1 \cup \cdots \cup S_m$. $\mathcal{B}$ first creates a normal ciphertext header and an encryption key by calling **Encrypt**(S^*, PK). Let $CH'_S = (E'_1, \dots, E'_7, C'_1, \{C'_{2,j}\}, \{\text{tag}_{c,j}\})$ and EK' be the normal ciphertext header and the encryption key under random exponents s'_1, s'_2, t'. It first modifies the normal ciphertext header to a semi-functional one by implicitly setting $s_1 = s'_1 - c_2$, $s_2 = s'_2 + c_1 + c_2$, and $s_3 = c_3$. The semi-functional challenge ciphertext header CH^* is described as follows:

$$
\begin{aligned}
&E_1 = E'_1 \cdot (g^{c_1})^b,\ E_2 = E'_2 \cdot (f^{c_2})^{-b},\ E_3 = E'_3 \cdot (f^{c_2})^{-1},\ E_4 = E'_4 \cdot (T)^b,\\
&E_5 = E'_5 \cdot T,\ E_6 = E'_6 \cdot (g^{c_1})^{v'} (f^{c_2})^{-v'_1} (T)^{v'_2},\ E_7 = E'_7 \cdot ((g^{c_1})^{v'} (f^{c_2})^{-v'_1} (T)^{v'_2})^b,\\
&C_1 = C'_1,\ \{C_{2,j} = C'_{2,j}\},\ \{\text{tag}_{c,j}\}.
\end{aligned}
$$

Next, it sets $EK_0 = EK' \cdot (e(g^{c_1}, f) \cdot e(g, f^{c_2}))^{b\alpha}$ and $EK_1 = \Omega^{\tilde{s}}$ by choosing a random exponent $\tilde{s} \in \mathbb{Z}_p$. It flips a random coin γ internally, and gives the tuple (CH^*, EK_γ) to $\mathcal{A}$. If $T = d^{c_1+c_2}$, then $\mathcal{B}$ is playing $\mathbf{G}_0$. Otherwise, it is playing $\mathbf{G}_1$. $\mathcal{A}$ outputs a guess γ'. If $\gamma = \gamma'$, it outputs 0. Otherwise, it outputs 1. □

Lemma 4.3. *If the DLIN assumption holds, then no polynomial-time adversary can distinguish between $\mathbf{G}_{1,k-1}$ and $\mathbf{G}_{1,k}$ with a non-negligible advantage.*

Proof. Suppose there exists an adversary $\mathcal{A}$ that distinguishes between $\mathbf{G}_{1,k-1}$ and $\mathbf{G}_{1,k}$ with a non-negligible advantage. A simulator $\mathcal{B}$ that solves the DLIN assumption using $\mathcal{A}$ is given: a challenge tuple $D = ((p, \mathbb{G}, \mathbb{G}_T, e), g, f, d, g^{c_1}, f^{c_2})$ and T where $T = d^{c_1+c_2}$ or $T = d^{c_1+c_2+c_3}$. Then $\mathcal{B}$ that interacts with $\mathcal{A}$ is described as follows: $\mathcal{B}$ first chooses random exponents $a_1, a_2, v', v'_1, v'_2, h'_1, \dots, h'_m, u'_1, \dots, u'_m, w', B_1, \dots, B_m, A_1, \dots, A_m, \alpha \in \mathbb{Z}_p$ and sets $v = d^{-a_1 a_2}, v_1 = d^{a_2} g^{v'_1}, v_2 = d^{a_1} g^{v'_2}$. It sets the master key as $MK = (g^{a_1 \alpha}, g^{-\alpha}, v, v_1, v_2)$ and publishes the public key PK as

$$
\begin{aligned}
&g,\ g^{a_1},\ g^{a_2},\ g^b = f,\ g^{a_1 b} = f^{a_1},\ g^{a_2 b} = f^{a_2},\ v v_1^{a_1} = g^{v'_1 a_1},\ v v_2^{a_2} = g^{v'_2 a_2},\\
&(v v_1^{a_1})^b = f^{v'_1 a_1},\ (v v_2^{a_2})^b = f^{v'_2 a_2},\ h_1 = g^{h'_1} f^{-B_1}, \dots, h_m = g^{h'_m} f^{-B_m},\\
&u_1 = g^{u'_1} f^{-A_1}, \dots, u_m = g^{u'_m} f^{-A_m},\ w = g^{w'} f,\ \Omega = e(g^{a_1}, f)^\alpha.
\end{aligned}
$$

$\mathcal{A}$ adaptively requests a private key query for an index d. If this is a ρ-th private key query for an index $d = (d_x, d_y)$, then $\mathcal{B}$ handles this query as follows:

- Case $\rho < k$: It first creates a normal private key by choosing random values $\text{tag}_k, \{z_i\}_{1 \le i \ne d_x \le m} \in \mathbb{Z}_p$ since it knows MK. Next, it converts the normal private key to a semi-functional one since it knows a_1 and a_2.
- Case $\rho = k$: It first creates a normal private key $SK'_d = (D'_1, \dots, D'_7, K'_1, \{K'_{2,i}\}, \text{tag}_k, \{z_i\})$ by setting $\text{tag}_k = B_{d_y} + A_{d_x}$, $\{z_i = A_i\}_{1 \le i \ne d_x \le m}$ since it knows MK. Let r'_1, r'_2, r'_3, r'_4 be the random exponents used in the normal private key. Next, it modifies the normal private key by implicitly setting $r_1 = r'_1 + c_1$, $r_2 = r'_2 + c_2$, $r_3 = r'_3 - c_2 v'_1$, $r_4 = r'_4 - c_2 v'_2$ and $r_5 = c_3$. The modified private key is described as follows:

  $$
  \begin{aligned}
  &D_1 = D'_1 \cdot (T)^{-a_1 a_2},\ D_2 = D'_2 \cdot (T)^{a_2} (g^{c_1})^{v'_1},\ D_3 = D'_3 \cdot (f^{c_2})^{v'_1},\ D_4 = D'_4 \cdot (T)^{a_1} (g^{c_1})^{v'_2},\\
  &D_5 = D'_5 \cdot (f^{c_2})^{v'_2},\ D_6 = D'_6 \cdot (f^{c_2})^{-1},\ D_7 = D'_7 \cdot (g^{c_1})^{-1},\\
  &K_1 = K'_1 \cdot (g^{c_1})^{h'_{d_y} + u'_{d_x} + w' \text{tag}_k},\ \{K_{2,i} = K'_{2,i} \cdot (g^{c_1})^{u'_i + w' z_i}\},\ \text{tag}_k, \{z_i\}.
  \end{aligned}
  $$

  If $T = d^{c_1+c_2}$, then $\mathcal{B}$ is playing $\mathbf{G}_{1,k-1}$. Otherwise, it is playing $\mathbf{G}_{1,k}$. Note that $\text{tag}_k = B_{d_y} + A_{d_x}$ and $z_j = A_j$ enables the cancellation of f^{c_1}.

- Case $\rho > k$: It creates a normal private key by choosing random values $\text{tag}_k, \{z_i\}_{1 \le i \ne d_x \le m}$ since it knows MK.

In the challenge step, $\mathcal{A}$ submits a challenge receiver set $S^* = S_1 \cup \cdots \cup S_m$. $\mathcal{B}$ first creates a normal ciphertext header by setting $\{\text{tag}_{c,j} = B_j + \sum_{i \in S_j} A_i\}_{1 \le j \le m}$. Let $CH' = (E'_1, \ldots, E'_7, C'_1, \{C'_{2,j}\}, \{\text{tag}_{c,j}\})$ and EK' be the normal ciphertext header and the encryption key under random exponents s'_1, s'_2, t'. It selects a random $s_3 \in \mathbb{Z}_p$ and modifies this ciphertext header by implicitly setting $t = t' + \log(d) a_1 a_2 s_3$. The modified semi-functional ciphertext header CH^* is described as follows:

$$
\begin{aligned}
&E_1 = E'_1,\ E_2 = E'_2,\ E_3 = E'_3,\ E_4 = E'_4 \cdot (f)^{a_2 s_3}, \\
&E_5 = E'_5 \cdot g^{a_2 s_3},\ E_6 = E'_6 \cdot v_2^{a_2 s_3},\ E_7 = E'_7 \cdot (d)^{-w' a_1 a_2 s_3} (f)^{a_2 v'_2 s_3}, \\
&C_1 = C'_1 \cdot (d)^{a_1 a_2 s_3},\ \big\{C_{2,j} = C'_{2,j} \cdot (d^{h'_j + \sum_{i \in S_j} u'_i + \text{tag}_{c,j} w'})^{a_1 a_2 s_3}\big\},\ \{\text{tag}_{c,j}\}.
\end{aligned}
$$

Next, it sets $EK_0 = EK'$ and $EK_1 = \Omega^{\tilde{s}}$ by choosing a random exponent $\tilde{s} \in \mathbb{Z}_p$. It flips a random coin γ internally, and gives the tuple (CH^*, EK_γ) to $\mathcal{A}$. Note that it can create the semi-functional ciphertext header since t and $\{\text{tag}_{c,j}\}$ enable the cancellation of $f^{\log(d)}$. $\mathcal{A}$ outputs a guess γ'. If $\gamma = \gamma'$, it outputs 0. Otherwise, it outputs 1.

The paradox of dual system encryption is solved since $\text{tag}_{c,j}$ of the ciphertext header and tag'_k derived from the i-th private key are the same if $d = (d_x, d_y)$ is a member of the subset S^*. Additionally, the adversary cannot detect any relationship between $\text{tag}_{c,j}$ of the ciphertext and tag_k of the i-th private key since the function $B_j + \sum_{i \in S_j} A_i$ is a pairwise independent function. □

Lemma 4.4. *If the DBDH assumption holds, then no polynomial-time adversary can distinguish between* $\mathbf{G}_2$ *and* $\mathbf{G}_3$ *with a non-negligible advantage.*

Proof. Suppose there exists an adversary $\mathcal{A}$ that distinguishes between $\mathbf{G}_2$ and $\mathbf{G}_3$ with a non-negligible advantage. A simulator $\mathcal{B}$ that solves the DBDH assumption using $\mathcal{A}$ is given: a challenge tuple $D = ((p, \mathbb{G}, \mathbb{G}_T, e),\ g, g^{c_1}, g^{c_2}, g^{c_3})$ and T where $T = e(g,g)^{c_1 c_2 c_3}$ or $T = e(g,g)^{c_4}$. Then $\mathcal{B}$ that interacts with $\mathcal{A}$ is described as follows: $\mathcal{B}$ first chooses random exponents $a_1, b, v', v'_1, v'_2 \in \mathbb{Z}_p$ and random elements $h_1, \ldots, h_m, u_1, \ldots, u_m, w \in \mathbb{G}$. It sets $v = g^{v'}, v_1 = g^{v'_1}, v_2 = g^{v'_2}$ and publishes the public key PK by implicitly setting $\alpha = c_1 c_2$ as

$$
\begin{aligned}
&g,\ g^{a_1},\ g^{a_2} = g^{c_2},\ g^b,\ g^{a_1 b},\ g^{a_2 b} = (g^{c_2})^b,\ v v_1^{a_1},\ v v_2^{a_2} = v (g^{c_2})^{v'_2}, \\
&(v v_1^{a_1})^b,\ (v v_2^{a_2})^b = (v (g^{c_2})^{v'_2})^b,\ h_1, \ldots, h_m,\ u_1, \ldots, u_m,\ w,\ \Omega = e(g^{c_1}, g^{c_2})^{a_1 b}.
\end{aligned}
$$

$\mathcal{A}$ adaptively requests a private key query for an index d. To response the query for an index $d = (d_x, d_y)$, $\mathcal{B}$ first selects random exponents $r_1, r_2, r_3, r_4, r'_5 \in \mathbb{Z}_p$ and random values $\text{tag}_k, \{z_i\}_{1 \le i \ne d_x \le m} \in \mathbb{Z}_p$. Next, it implicitly sets $r_5 = c_1 + r'_5$ and creates a semi-functional private key as

$$
\begin{aligned}
&D_1 = v^{r_1 + r_2} (g^{c_2})^{-a_1 r'_5},\ D_2 = v_1^{r_1 + r_2} g^{r_3} (g^{c_2})^{r'_5},\ D_3 = (g^{-b})^{r_3},\ D_4 = v_2^{r_1 + r_2} g^{r_4} (g^{c_1})^{a_1} g^{a_1 r'_5}, \\
&D_5 = (g^{-b})^{r_4},\ D_6 = (g^{-b})^{r_2},\ D_7 = (g^{-1})^{r_1}, \\
&K_1 = (h_{d_y} u_{d_x})^{r_1} w^{\text{tag}_k r_1},\ \{K_{2,i} = u_i^{r_1} w^{z_i r_1}\}_{1 \le i \ne d_x \le m},\ \text{tag}_k, \{z_i\}.
\end{aligned}
$$

Note that it can only create a semi-functional private key since $r_5 = c_1 + r'_5$ enables the cancellation of $g^{c_1 c_2}$. In the challenge step, $\mathcal{A}$ submits a challenge receiver set $S^* = S_1 \cup \cdots \cup S_m$. $\mathcal{B}$ chooses random exponents

$s_1, s'_3, t \in \mathbb{Z}_p$ and random values $\{\text{tag}_{c,j}\}_{1\le j\le m} \in \mathbb{Z}_p$. Next, it implicitly sets $s_2 = c_3, s_3 = -c_3 + s'_3$ and creates a semi-functional ciphertext header CH^* as

$$E_1 = g^{bs_1}(g^{c_3})^b,\ E_2 = g^{a_1 bs_1},\ E_3 = g^{a_1 s_1},\ E_4 = (g^{c_2})^{bs'_3},$$
$$E_5 = (g^{c_2})^{s'_3},\ E_6 = (vv_1^{a_1})^{s_1}(g^{c_3})^{v'}(g^{c_2})^{v'_2 s'_3},\ E_7 = (vv_1^{a_1})^{bs_1}(g^{c_3})^{v'b}(g^{c_2})^{v'_2 bs'_3} w^{-t},$$
$$C_1 = g^t,\ \Big\{C_{2,j} = (h_j \prod_{i\in S_j} u_i)^t w^{\text{tag}_{c,j}t}\Big\}_{1\le j\le m},\ \{\text{tag}_{c,j}\}.$$

Next, it sets $EK_0 = (T)^{a_1 b}$ and $EK_1 = \Omega^{\tilde{s}}$ by choosing a random exponent $\tilde{s} \in \mathbb{Z}_p$. It flips a random coin γ internally, and gives the tuple (CH^*, EK_γ) to $\mathcal{A}$. If $T = e(g,g)^{c_1c_2c_3}$, then $\mathcal{B}$ is playing $\mathbf{G}_2$. Otherwise, it is playing $\mathbf{G}_3$. Note that it can only create a semi-functional ciphertext header since $s_3 = -c_3 + s'_3$ enables the cancellation of $g^{c_2c_3}$. $\mathcal{A}$ outputs a guess γ'. If $\gamma = \gamma'$, it outputs 0. Otherwise, it outputs 1. □

5 Asymmetric Construction

In this section, we present an efficient PKBE scheme based on asymmetric bilinear groups of prime order and prove its adaptive security under three simple assumptions.

5.1 Asymmetric Bilinear Groups

Let $\mathbb{G}, \hat{\mathbb{G}}$, and $\mathbb{G}_T$ be multiplicative cyclic groups of prime p order where $\mathbb{G} \neq \hat{\mathbb{G}}$. Let $g, \hat{g}$ be generators of $\mathbb{G}, \hat{\mathbb{G}}$, respectively. The bilinear map $e : \mathbb{G} \times \hat{\mathbb{G}} \to \mathbb{G}_T$ has the following properties:

1. Bilinearity: $\forall u \in \mathbb{G}, \forall \hat{v} \in \hat{\mathbb{G}}$ and $\forall a, b \in \mathbb{Z}_p$, $e(u^a, \hat{v}^b) = e(u, \hat{v})^{ab}$.
2. Non-degeneracy: $\exists g, \hat{g}$ such that $e(g, \hat{g})$ has order p, that is, $e(g, \hat{g})$ is a generator of $\mathbb{G}_T$.

We say that $\mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T$ are asymmetric bilinear groups if the group operations in $\mathbb{G}, \hat{\mathbb{G}}$, and $\mathbb{G}_T$ as well as the bilinear map e are all efficiently computable.

5.2 Complexity Assumptions

We introduce three simple assumptions under asymmetric bilinear groups of prime order.

Assumption 5.1 (decisional eXternal Diffie-Hellman, XDH [5]). *Let $(p, \mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T, e)$ be a description of the asymmetric bilinear group of prime order p. The XDH assumption is that if the challenge values*

$$D = ((p, \mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T, e),\ g, g^a, g^b, \hat{g}) \text{ and } T$$

are given, no PPT algorithm $\mathcal{A}$ can distinguish $T = T_0 = g^{ab}$ from $T = T_1 = g^c$ with more than a negligible probability. The advantage of $\mathcal{A}$ is defined as $\mathbf{Adv}_{\mathcal{A}}^{XDH} = \big|\Pr[\mathcal{A}(D, T_0) = 1] - \Pr[\mathcal{A}(D, T_1) = 1]\big|$ where the probability is taken over the random choices of $a, b, c \in \mathbb{Z}_p$.

Assumption 5.2 (Asymmetric 3-Party Diffie-Hellman, A3DH). *Let $(p, \mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T, e)$ be a description of the asymmetric bilinear group of prime order p. The A3DH assumption is that if the challenge values*

$$D = ((p, \mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T, e),\ g, g^a, g^b, \hat{g}, \hat{g}^a, \hat{g}^{ab}, \hat{g}^c) \text{ and } T$$

are given, no PPT algorithm $\mathcal{A}$ can distinguish $T = T_0 = \hat{g}^{abc}$ from $T = T_1 = \hat{g}^d$ with more than a negligible probability. The advantage of $\mathcal{A}$ is defined as $\mathbf{Adv}_{\mathcal{A}}^{A3DH} = \big|\Pr[\mathcal{A}(D, T_0) = 1] - \Pr[\mathcal{A}(D, T_1) = 1]\big|$ where the probability is taken over the random choices of $a, b, c, d \in \mathbb{Z}_p$.

Assumption 5.3 (Decisional Bilinear Diffie-Hellman, DBDH). *Let $(p, \mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T, e)$ be a description of the asymmetric bilinear group of prime order p. The DBDH assumption is that if the challenge values*

$$D = ((p, \mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T, e),\ g, g^a, g^b, g^c, \hat{g}, \hat{g}^a, \hat{g}^b, \hat{g}^c) \text{ and } T$$

are given, no PPT algorithm $\mathcal{A}$ can distinguish $T = T_0 = e(g, \hat{g})^{abc}$ from $T = T_1 = e(g, \hat{g})^d$ with more than a negligible probability. The advantage of $\mathcal{A}$ is defined as $\boldsymbol{Adv}_{\mathcal{A}}^{DBDH} = \big|\Pr[\mathcal{A}(D, T_0) = 1] - \Pr[\mathcal{A}(D, T_1) = 1]\big|$ where the probability is taken over the random choices of $a, b, c, d \in \mathbb{Z}_p$.

5.3 Construction

Let N be the total number of users and $m = \lceil \sqrt{N} \rceil$. An index $d \in \{1, \ldots, N\}$ is represented as a position (d_x, d_y) in a $m \times m$ matrix where $d = (d_y - 1)m + d_x$ for some $1 \le d_y \le m$ and $1 \le d_x \le m$. Let S be a subset of $\{1, \ldots, N\}$, and define $S'_j = S \cap \{(j-1)m+1, \ldots, (j-1)m+m\}$ and $S_j = \{x - (j-1)m | x \in S'_j\} \subseteq \{1, \ldots, m\}$. A subset S is divided to subsets $S_1, \ldots, S_m$.

PKBE.Setup($1^\lambda, N$): This algorithm first generates the asymmetric bilinear groups $\mathbb{G}, \hat{\mathbb{G}}$ of prime order p of bit size $\Theta(\lambda)$. Let $g, \hat{g}$ be the generator of $\mathbb{G}, \hat{\mathbb{G}}$ respectively. Next, it chooses random exponents $a, v', v'_1, h'_1, \ldots, h'_m, u'_1, \ldots, u'_m, w', \alpha \in \mathbb{Z}_p$ and sets $v = g^{v'}, v_1 = g^{v'_1}, \{h_i = g^{h'_i}, u_i = g^{u'_i}\}_{1 \le i \le m}, w = g^{w'}, \hat{v} = \hat{g}^{v'}, \hat{v}_1 = \hat{g}^{v'_1}, \{\hat{h}_i = \hat{g}^{h'_i}, \hat{u}_i = \hat{g}^{u'_i}\}_{1 \le i \le m}, \hat{w} = \hat{g}^{w'}$. It outputs a master key $MK = (\hat{g}^\alpha, \hat{v}, \hat{v}_1, \{\hat{h}_i, \hat{u}_i\}_{1 \le i \le m}, \hat{w})$ and a public key as

$$PK = \Big(\ g,\ g^a,\ vv_1^a,\ h_1, \ldots, h_m,\ u_1, \ldots, u_m,\ w,\ \Omega = e(g, \hat{g})^\alpha\ \Big).$$

PKBE.KeyGen(d, MK, PK): This algorithm takes as input an index $d = (d_x, d_y)$, the master key MK, and the public key PK. It selects a random exponent $r_1 \in \mathbb{Z}_p$ and random values $\text{tag}_k, z_1, \ldots, z_m \in \mathbb{Z}_p$. It outputs a private key by implicitly including d as

$$SK_d = \Big(\ D_1 = \hat{g}^\alpha \hat{v}^{r_1},\ D_2 = \hat{v}_1^{r_1},\ D_3 = \hat{g}^{-r_1},$$
$$K_1 = (\hat{h}_{d_y} \hat{u}_{d_x})^{r_1} \hat{w}^{\text{tag}_k r_1},\ \big\{K_{2,i} = \hat{u}_i^{r_1} \hat{w}^{z_i r_1}\big\}_{1 \le i \ne d_x \le m},\ \text{tag}_k,\ \{z_i\}_{1 \le i \ne d_x \le m}\ \Big).$$

PKBE.Encrypt(S, PK): This algorithm takes as input a receiver set S that divided to subsets $S_1, \ldots, S_m$ and the public key PK. It first chooses random exponents $s_1, t \in \mathbb{Z}_p$ and random values $\text{tag}_{c,1}, \ldots, \text{tag}_{c,m} \in \mathbb{Z}_p$. It outputs a ciphertext header by implicitly including S as

$$CH_S = \Big(\ E_1 = g^{s_1},\ E_2 = (g^a)^{s_1},\ E_3 = (vv_1^a)^{s_1} w^{-t},$$
$$C_1 = g^t,\ \big\{C_{2,j} = (h_j \prod_{i \in S_j} u_i)^t w^{\text{tag}_{c,j} t}\big\}_{1 \le j \le m},\ \{\text{tag}_{c,j}\}_{1 \le j \le m}\ \Big)$$

and an encryption key $EK = \Omega^{s_1}$.

PKBE.Decrypt(CH_S, SK_d, PK): This algorithm takes as input a ciphertext header CH_S for a receiver set $S = S_1 \cup \cdots \cup S_m$ and a private key SK_d for an index $d = (d_x, d_y)$. If $d \notin S$, it outputs $\perp$. Otherwise it proceeds as follows:

1. It finds a subset S_{d_y} from the set S such that $d_x \in S_{d_y}$ and calculates $\text{tag}'_k = \text{tag}_k + \sum_{i \in S_{d_y} \setminus \{d_x\}} z_i$ from the private key.

2. If $\mathrm{tag}'_k \neq \mathrm{tag}_{c,d_y}$, then it outputs an encryption key as

$$EK = \prod_{i=1}^{3} e(E_i, D_i) \cdot \Big(e(C_1, K_1 \prod_{i \in S_{d_y} \setminus \{d_x\}} K_{2,i}) \cdot e(C_{2,d_y}, D_3) \Big)^{-1/(\mathrm{tag}'_k - \mathrm{tag}_{c,d_y})}.$$

Otherwise, it outputs $\perp$.

5.4 Correctness

Let $\mathrm{tag}'_k = \mathrm{tag}_k + \sum_{i \in S_{d_y} \setminus \{d_x\}} z_i$. If $\mathrm{tag}'_k \neq \mathrm{tag}_{c,d_y}$, then the correctness of the above PKBE scheme is easily verified as

$$\begin{aligned} &\Big(\prod_{i=1}^{3} e(E_i, D_i) \Big) \cdot \Big(e(C_1, K_1 \prod_{i \in S_{d_y} \setminus \{d_x\}} K_{2,i}) \cdot e(D_7, C_2) \Big)^{-\frac{1}{(\mathrm{tag}'_k - \mathrm{tag}_{c,d_y})}} \\ &= \Big(e(g^s, \hat{g}^{\alpha}) \cdot e(w^t, \hat{g}^{r_1}) \Big) \cdot e(g^t, \hat{w}^{r_1})^{-1} = e(g, \hat{g})^{\alpha s}. \end{aligned}$$

Note that we have $\mathrm{tag}'_k \neq \mathrm{tag}_{c,d_y}$ with $1 - 1/p$ probability since $\mathrm{tag}_k, \{z_i\}_{1 \leq i \neq d_x \leq m}, \{\mathrm{tag}_{c,j}\}_{1 \leq j \leq m}$ are randomly chosen in $\mathbb{Z}_p$.

5.5 Security Analysis

Theorem 5.4. *The above PKBE scheme is adaptively secure under a chosen ciphertext attack if the XDH, A3DH, and DBDH assumptions hold. That is, for any PPT adversary $\mathcal{A}$, there exists a PPT algorithm $\mathcal{B}$ such that*

$$\boldsymbol{Adv}_{\mathcal{A}}^{PKBE} \leq \boldsymbol{Adv}_{\mathcal{B}}^{XDH} + N\boldsymbol{Adv}_{\mathcal{B}}^{A3DH} + \boldsymbol{Adv}_{\mathcal{A}}^{DBDH}$$

where N is the number of total users in the system.

Proof. To prove the security of our scheme in dual system encryption, we first define the semi-functional private keys and ciphertext headers.

PKBE.KeyGenSF. This algorithm first creates a normal private key $SK'_d = (D'_1, D'_2, D'_3, K'_1, \{K'_{2,i}\}, \mathrm{tag}_k, \{z_i\})$. It chooses a random exponent $r_2 \in \mathbb{Z}_p$ and outputs a semi-functional private key as

$$SK_d = \Big(D_1 = D'_1 \cdot (\hat{g}^a)^{-r_2},\ D_2 = D'_2 \cdot \hat{g}^{r_2},\ D_3 = D'_3,\ K_1 = K'_1,\ \{K_{2,i} = K'_{2,i}\},\ \mathrm{tag}_k,\ \{z_i\} \Big).$$

PKBE.EncryptSF. This algorithm first creates a normal ciphertext header $CH'_S = (E'_1, E'_2, E'_3, C'_1, \{C'_{2,j}\}, \{\mathrm{tag}_{c,j}\})$ and an encryption key EK'. It chooses a random exponent $s_2 \in \mathbb{Z}_p$ and outputs a semi-functional ciphertext header as

$$CH_S = \Big(E_1 = E'_1,\ E_2 = E'_2 \cdot (g^a)^{s_2},\ E_3 = E'_3 \cdot (v_1^a)^{s_2},\ C_1 = C'_1,\ \{C_{2,j} = C'_{2,j}\},\ \{\mathrm{tag}_{c,j}\} \Big)$$

and an encryption key $EK = EK'$.

Note that if a semi-functional private key is used to decrypt a semi-functional ciphertext header, then the decryption algorithm will fail to produce a valid encryption key since it is multiplied by an additional term $e((g^a)^{s_2}, \hat{g}^{r_2})$.

The security proof also consists of the sequence of hybrid games that are defined in Theorem 4.1. From the following three lemmas, we have that it is hard to distinguish $\mathbf{G}_{i-1}$ from $\mathbf{G}_i$ under the given assumptions. This completes our proof. $\square$

Lemma 5.5. *If the XDH assumption in $\mathbb{G}$ holds, then no polynomial-time adversary can distinguish between $\mathbf{G}_0$ and $\mathbf{G}_1$ with a non-negligible advantage.*

Proof. Suppose there exists an adversary $\mathcal{A}$ that distinguishes between $\mathbf{G}_0$ and $\mathbf{G}_1$ with a non-negligible advantage. A simulator $\mathcal{B}$ that solves the XDH assumption in $\mathbb{G}$ using $\mathcal{A}$ is given: a challenge tuple $D = ((p, \mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T, e), g, g^{c_1}, g^{c_2}, \hat{g})$ and T where $T = g^{c_1 c_2}$ or $T = g^{c_1 c_2 + c_1 c_3}$. Then $\mathcal{B}$ that interacts with $\mathcal{A}$ is described as follows: $\mathcal{B}$ first chooses random exponents $v', v'_1, \{h'_i, u'_i\}_{1 \le i \le m}, w', \alpha \in \mathbb{Z}_p$ and sets $\{h_i = g^{h'_i}, u_i = g^{u'_i}\}_{1 \le i \le m}, w = g^{w'}, \hat{v} = \hat{g}^{v'}, \hat{v}_1 = \hat{g}^{v'_1}, \{\hat{h}_i = \hat{g}^{h'_i}, \hat{u}_i = \hat{g}^{u'_i}\}_{1 \le i \le m}, \hat{w} = \hat{g}^{w'}$. It sets the master key as $MK = (\hat{g}^\alpha, \hat{v}, \hat{v}_1, \{\hat{h}_i, \hat{u}_i\}_{1 \le i \le m}, \hat{w})$ and publishes the public key PK by implicitly setting $a = c_1$ as

$$g,\ g^a = g^{c_1},\ v v_1^a = g^{v'} (g^{c_1})^{v'_1},\ h_1, \ldots, h_m,\ u_1, \ldots, u_m,\ w,\ \Omega = e(g, \hat{g})^\alpha.$$

$\mathcal{A}$ adaptively requests a private key query for an index d. To response this query, $\mathcal{B}$ simply runs the key generation algorithm to create a normal private key using the master key. Note that it can only create the normal private keys since it does not know a. In the challenge step, $\mathcal{A}$ submits a challenge receiver set $S^* = S_1 \cup \cdots \cup S_m$. $\mathcal{B}$ first creates a normal ciphertext by calling $\mathbf{Encrypt}(S^*, PK)$. Let $CH'_S = (E'_1, E'_2, E'_3, C'_1, \{C'_{2,j}\}, \{\text{tag}_{c,j}\})$ and EK' be the normal ciphertext header and the encryption key under random exponents $s'_1, t' \in \mathbb{Z}_p$. It modifies the ciphertext header by implicitly setting $s_1 = s'_1 + c_2$ and $s_2 = c_3$. The modified semi-functional ciphertext header CH^* is described as follows:

$$E_1 = E'_1 \cdot g^{c_2},\ E_2 = E'_2 \cdot T,\ E_3 = E'_3 \cdot (g^{c_2})^{v'} (T)^{v'_1},\ C_1 = C'_1,\ \{C_{2,j} = C'_{2,j}\},\ \{\text{tag}_{c,j}\}.$$

Next, it sets $EK_0 = EK'$ and $EK_1 = \Omega^{\tilde{s}}$ by choosing a random exponent $\tilde{s} \in \mathbb{Z}_p$. It flips a random coin γ internally, and gives the tuple (CH^*, EK_γ) to $\mathcal{A}$. If $T = g^{c_1 c_2}$, then $\mathcal{B}$ is playing $\mathbf{G}_0$. Otherwise, it is playing $\mathbf{G}_1$. $\mathcal{A}$ outputs a guess γ'. If $\gamma = \gamma'$, it outputs 0. Otherwise, it outputs 1. $\square$

Lemma 5.6. *If the A3DH assumption holds, then no polynomial-time adversary can distinguish between $\mathbf{G}_{1,k-1}$ and $\mathbf{G}_{1,k}$ with a non-negligible advantage.*

Proof. Suppose there exists an adversary $\mathcal{A}$ that distinguishes between $\mathbf{G}_{1,k-1}$ and $\mathbf{G}_{1,k}$ with a non-negligible advantage. A simulator $\mathcal{B}$ that solves the A3DH assumption in $\hat{\mathbb{G}}$ using $\mathcal{A}$ is given: a challenge tuple $D = ((p, \mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T, e), g, g^{c_1}, g^{c_2}, \hat{g}, \hat{g}^{c_1}, \hat{g}^{c_1 c_2}, \hat{g}^{c_3})$ and T where $T = \hat{g}^{c_1 c_2 c_3}$ or $T = \hat{g}^{c_1 c_2 c_3 + c_4}$. Then $\mathcal{B}$ that interacts with $\mathcal{A}$ is described as follows: $\mathcal{B}$ first chooses random exponents $a, v', v'_1, h'_1, \ldots, h'_m, u'_1, \ldots, u'_m, w', B_1, \ldots, B_m, A_1, \ldots, A_m, \alpha \in \mathbb{Z}_p$ and sets $\{h_i = g^{h'_i} (g^{c_1})^{-B_i}, u_i = g^{u'_i} (g^{c_1})^{-A_i}\}, w = g^{w'} g^{c_1}, \hat{v} = (\hat{g}^{c_1 c_2})^{-a}, \hat{v}_1 = (\hat{g}^{c_1 c_2}) \hat{g}^{v'_1}, \{\hat{h}_i = \hat{g}^{h'_i} (\hat{g}^{c_1})^{-B_i}, \hat{u}_i = \hat{g}^{u'_i} (\hat{g}^{c_1})^{-A_i}\}, \hat{w} = \hat{g}^{w'} \hat{g}^{c_1}$. It sets the master key $MK = (\hat{g}^\alpha, \hat{v}, \hat{v}_1, \{\hat{h}_i, \hat{u}_i\}, \hat{w})$ and publishes the public key PK as

$$g,\ g^a,\ v v_1^a = g^{v'_1 a},\ h_1, \ldots, h_m,\ u_1, \ldots, u_m,\ w,\ \Omega = e(g, \hat{g})^\alpha.$$

$\mathcal{A}$ adaptively requests a private key query for an index d. If this is a ρ-th private key query for an index $d = (d_x, d_y)$, then $\mathcal{B}$ handles this query as follows:

- Case $\rho < k$: It first creates a normal private key by choosing random values $\text{tag}_k, \{z_i\}_{1 \le i \ne d_x \le m} \in \mathbb{Z}_n$ since it knows MK. Next, it converts the normal private key to a semi-functional one since it knows a.

- Case $\rho = k$: It first creates a normal private key $SK'_d = (D'_1, D'_2, D'_3, K'_1, \{K'_{2,i}\}, \text{tag}_k, \{z_i\})$ by setting $\text{tag}_k = B_{d_y} + A_{d_x}, \{z_i = A_i\}_{1 \le i \ne d_x \le m}$ since it knows MK. Let r'_1 be the random exponent used in the normal private key. Next, it modifies the private key by implicitly setting $r_1 = r'_1 + c_3$ and $r_2 = c_3$. The modified private key is described as follows:

$$D_1 = D'_1 \cdot (T)^{-a},\ D_2 = D'_2 \cdot T(\hat{g}^{c_3})^{v'_1},\ D_3 = D'_3 \cdot (\hat{g}^{c_3})^{-1},$$
$$K_1 = K'_1 \cdot (\hat{g}^{c_3})^{h'_{d_y} + u'_{d_x} + w' \text{tag}_k},\ \{K_{2,i} = K'_{2,i} \cdot (\hat{g}^{c_3})^{u'_i + w' z_i}\},\ \text{tag}_k, \{z_i\}.$$

If $T = \hat{g}^{c_1 c_2 c_3}$, then $\mathcal{B}$ is playing $\mathbf{G}_{1,k-1}$. Otherwise, it is playing $\mathbf{G}_{1,k}$. Note that $\text{tag}_k = B_{d_y} + A_{d_x}$ and $z_j = A_j$ enables the cancellation of $\hat{g}^{c_1 c_3}$.

- Case $\rho > k$: It creates a normal private key by choosing random values $\text{tag}_k, \{z_i\}_{1 \le i \ne d_x \le m} \in \mathbb{Z}_p$ since it knows MK.

In the challenge step, $\mathcal{A}$ submits a challenge receiver set $S^* = S_1 \cup \cdots \cup S_m$. $\mathcal{B}$ first creates a normal ciphertext by setting $\{\text{tag}_{c,j} = B_j + \sum_{i \in S_j} A_i\}_{1 \le j \le m}$. Let $CH'_S = (E'_1, E'_2, E'_3, C'_1, \{C'_{2,j}\}, \{\text{tag}_{c,j}\})$ and EK' be the normal ciphertext header and the encryption key under random exponents $s'_1, t' \in \mathbb{Z}_p$. It selects a random exponent $s_2 \in \mathbb{Z}_p$ and modifies this ciphertext header by implicitly setting $t = t' + c_2 a s_2$. The modified semi-functional ciphertext header CH^* is described as follows:

$$E_1 = E'_1,\ E_2 = E'_2 \cdot (g^a)^{s_2},\ E_3 = E'_3 \cdot (g^{c_2})^{-w' a s_2} g^{v'_1 a s_2},$$
$$C_1 = C'_1 \cdot (g^{c_2})^{a s_2},\ \big\{C_{2,j} = C'_{2,j} \cdot (g^{c_2})^{a s_2 (h'_j + \sum_{i \in S_j} u'_i + \text{tag}_{c,j} w')}\big\}_{1 \le j \le m},\ \{\text{tag}_{c,j}\}.$$

Next, it sets $EK_0 = EK'$ and $EK_1 = \Omega^{\tilde{s}}$ by choosing a random exponent $\tilde{s} \in \mathbb{Z}_p$. It flips a random coin γ internally, and gives the tuple (CH^*, EK_γ) to $\mathcal{A}$. Note that it can create a semi-functional ciphertext header since $t = t' + c_2 a s_2$ and $\{\text{tag}_{c,j} = B_j + \sum_{i \in S_j} A_i\}$ enable the cancellation of $g^{c_1 c_2}$. $\mathcal{A}$ outputs a guess γ'. If $\gamma = \gamma'$, it outputs 0. Otherwise, it outputs 1.

The paradox of dual system encryption is solved since $\text{tag}_{c,j}$ of the ciphertext header and tag'_k derived from the i-th private key are the same if $d = (d_x, d_y)$ is a member of the subset S^*. Additionally, the adversary cannot detect any relationship between $\text{tag}_{c,j}$ of the ciphertext and tag_k of the i-th private key since the function $B_j + \sum_{i \in S_j} A_i$ is a pairwise independent function. □

Lemma 5.7. *If the DBDH assumption holds, then no polynomial-time adversary can distinguish between* $\mathbf{G}_2$ *and* $\mathbf{G}_3$ *with a non-negligible advantage.*

Proof. Suppose there exists an adversary $\mathcal{A}$ that distinguishes between $\mathbf{G}_2$ and $\mathbf{G}_3$ with a non-negligible advantage. A simulator $\mathcal{B}$ that solves the DBDH assumption using $\mathcal{A}$ is given: a challenge tuple $D = ((p, \mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T, e),\ g, g^{c_1}, g^{c_2}, g^{c_3}, \hat{g}, \hat{g}^{c_1}, \hat{g}^{c_2}, \hat{g}^{c_3})$ and T where $T = e(g, \hat{g})^{c_1 c_2 c_3}$ or $T = e(g, \hat{g})^{c_4}$. Then $\mathcal{B}$ that interacts with $\mathcal{A}$ is described as follows: $\mathcal{B}$ first chooses random exponents $v', v'_1, \{h'_i, u'_i\}_{1 \le i \le m}, w' \in \mathbb{Z}_p$ and sets $\{h_i = g^{h'_i}, u_i = g^{u'_i}\}_{1 \le i \le m}, w = g^{w'}, \hat{v} = \hat{g}^{v'}, \hat{v}_1 = \hat{g}^{v'_1}, \{\hat{h}_i = \hat{g}^{h'_i}, \hat{u}_i = \hat{g}^{u'_i}\}_{1 \le i \le m}, \hat{w} = \hat{g}^{w'}$. It implicitly sets $a = c_2, \alpha = c_1 c_2$ and publishes the public key PK as

$$g,\ g^a = g^{c_2},\ v v_1^a = v(g^{c_2})^{v'_1},\ h_1,\ \ldots,\ h_m,\ u_1,\ \ldots,\ u_m,\ w,\ \Omega = e(g^{c_1}, \hat{g}^{c_2}).$$

Table 2: The size of groups and the cost of operations in bilinear groups

Type	Len($\mathbb{Z}_p$) (bits)	Len($\mathbb{G}$) (bits)	Len($\hat{\mathbb{G}}$) (bits)	Len($\mathbb{G}_T$) (bits)	Exp($\mathbb{G}$) (ms)	Exp($\hat{\mathbb{G}}$) (ms)	Exp($\mathbb{G}_T$) (ms)	Pair (ms)
Symmetric	160	512	-	1024	4.2	-	0.8	6.2
Asymmetric	168	176	528	1056	1.6	20.3	5.5	15.6

Symmetric = supersingular curve, Asymmetric = MNT curve

Len($-$) = the bit size of group elements, Exp($-$) = exponentiation, Pair = pairing

$\mathcal{A}$ adaptively requests a private key query for an index d. To response this query for an index $d = (d_x, d_y)$, $\mathcal{B}$ selects random exponents $r_1, r_2' \in \mathbb{Z}_p$ and random values $\text{tag}_k, \{z_i\}_{1 \leq i \neq d_x \leq m} \in \mathbb{Z}_p$. Next, it implicitly sets $r_2 = c_1 + r_2'$ and creates a semi-functional private key as

$$D_1 = \hat{v}^{r_1}(\hat{g}^{c_2})^{-r_2'},\ D_2 = \hat{v}_1^{r_1}(\hat{g}^{c_1})\hat{g}^{r_2'},\ D_3 = \hat{g}^{-r_1},$$
$$K_1 = (\hat{h}_{d_y}\hat{u}_{d_x})^{r_1}\hat{w}^{\text{tag}_k r_1},\ \{K_{2,i} = \hat{u}_i^{r_1}\hat{w}^{z_i r_1}\}_{1 \leq i \neq d_x \leq m},\ \text{tag}_k,\ \{z_i\}.$$

Note that it can only create a semi-functional private key since $r_2 = c_1 + r_2'$ enables the cancellation of $\hat{g}^{c_1 c_2}$. In the challenge step, $\mathcal{A}$ submits a challenge receiver set $S^* = S_1 \cup \cdots \cup S_m$. It chooses random exponents $s_1, s_2', t \in \mathbb{Z}_p$ and random values $\{\text{tag}_{c,j}\}_{1 \leq j \leq m} \in \mathbb{Z}_p$. Next, it implicitly sets $s_1 = c_3, s_2 = -c_3 + s_2'$ and creates a semi-functional ciphertext header CH^* as

$$E_1 = g^{c_3},\ E_2 = (g^{c_2})^{s_2'},\ E_3 = (g^{c_3})^{v'}(g^{c_2})^{v_1' s_2'} w^{-t},$$
$$C_1 = g^t,\ \{C_{2,j} = (h_j \prod_{i \in S_j} u_i)^t w^{\text{tag}_{c,j} t}\}_{1 \leq j \leq m},\ \{\text{tag}_{c,j}\}.$$

Next, it sets $EK_0 = T$ and $EK_1 = \Omega^{\tilde{s}}$ by choosing a random exponent $\tilde{s} \in \mathbb{Z}_p$. It flips a random coin γ internally, and gives the tuple (CH^*, EK_γ) to $\mathcal{A}$. If $T = e(g, \hat{g})^{c_1 c_2 c_3}$, then $\mathcal{B}$ is playing $\mathbf{G}_2$. Otherwise, it is playing $\mathbf{G}_3$. Note that it can only create a semi-functional ciphertext header since $s_2 = -c_3 + s_2'$ enables the cancellation of $g^{c_2 c_3}$. $\mathcal{A}$ outputs a guess γ'. If $\gamma = \gamma'$, it outputs 0. Otherwise, it outputs 1. $\square$

6 Efficiency Comparison

In this section, we compare the efficiency of our schemes with that of other schemes. For symmetric bilinear groups that achieves the 80-bit security level, we select the supersingular curve with embedding degree 2 for large prime characteristic. For asymmetric bilinear groups that achieves the 80-bit security level, we select the Miyaji-Nakabayashi-Takano (MNT) curve with embedding degree 6. To compare the performance of schemes, we used the Pairing Based Cryptography (PBC) library of Lynn [22] to measure the cost of each operations in these bilinear groups[1]. The detailed information of these bilinear groups is given in Table 2. Note that we can assume that the cost of 160 multiplications is approximately equal to the cost of one exponentiation.

Symmetric Bilinear Groups. In symmetric bilinear groups, we compare our PKBE scheme with the PKBE scheme of Waters [29], the PKRE scheme of Lewko et al. [19], and the AugBE scheme of Garg et al. [16].

[1] We measured the cost of each operations under a laptop computer with an Intel Core i5-460M 2.53 GHz CPU.

Table 3: The efficiency comparison of schemes in symmetric bilinear groups

Scheme	PK Size (kbits)	SK Size (kbits)	CT Size (kbits)	KeyGen (sec)	Encrypt (sec)	Decrypt (sec)
Waters [28]	$5.1 * 10^5$	$5.1 * 10^5$	5.6	$4.2 * 10^3$	26.3	26.3
LSW [19]	7.2	4.1	$1.0 * 10^4$	0.05	$1.3 * 10^2$	$1.5 * 10^2$
GKSW [16]	$2.6 * 10^3$	$0.5 * 10^3$	$8.2 * 10^3$	4.2	98.4	0.08
Ours	$1.0 * 10^3$	$0.7 * 10^3$	$0.5 * 10^3$	8.4	34.6	0.08

$N = 10^6$ = the number of total users, $r = 10^4$ = the number of revoked users

Table 4: The efficiency comparison of schemes in asymmetric bilinear groups

Scheme	PK Size (kbits)	SK Size (kbits)	CT Size (kbits)	KeyGen (sec)	Encrypt (sec)	Decrypt (sec)
GKSW [16]	$1.9 * 10^3$	$0.5 * 10^3$	$4.2 * 10^3$	20.3	146.9	0.22
PRL [27]	$2.5 * 10^3$	$0.5 * 10^3$	$2.8 * 10^3$	20.3	123.4	0.19
Ours	$0.4 * 10^3$	$0.7 * 10^3$	$0.3 * 10^3$	40.6	13.2	0.21

$N = 10^6$ = the number of total users, $r = 10^4$ = the number of revoked users

Suppose that the number of total users N is 10^6 and the number of revoked users r is 10^4. To measure the performance of each algorithms, we assume that these algorithms are naively implemented by just using the basic operations in Table 2. The detailed efficiency comparison of these schemes in symmetric bilinear groups is given in Table 3. As mentioned, the PKBE scheme of Waters is not appropriate for the system with the large number of total users N since the public key size, the private key size, and the cost of the key generation algorithm are huge compared with other schemes. The PKRE scheme of Lewko et al. is also not appropriate for the system with the large number of revoked users since the ciphertext size, the cost of the encryption and decryption algorithms are proportional to the value r. Our scheme and the AugBE scheme of Garg et al. provide the reasonable size of public keys, private keys, and ciphertexts. Additionally, the cost of these algorithms in these two schemes is constant. However, the ciphertext size of our scheme is 94% shorter and the encryption algorithm of our scheme is 2.8 times faster than those of the AugBE scheme of Garg et al.

Asymmetric Bilinear Groups. In asymmetric bilinear groups, we compare our PKBE scheme with AugBE schemes of Garg et al. [16] and Park et al. [27]. The main advantage of asymmetric bilinear groups is that it provide shorter representation in $\mathbb{G}$ and efficient exponentiations in $\mathbb{G}$. The detailed efficiency comparison of these schemes in asymmetric bilinear groups is given in Table 4. The AugBE scheme of Park et al. performs better than the AugBE scheme of Garg et al. in terms of ciphertext size and encryption cost. However, the ciphertext size of our PKBE scheme is 90% shorter and the encryption algorithm of our PKBE scheme is 9.3 times faster than those of the AugBE scheme of Park et al.

7 Conclusion

In this paper, we proposed efficient PKBE schemes with sub-linear size of public keys, private keys, and ciphertexts, and proved their adaptive security under standard (or simple) assumptions. To enable our schemes, we first devised a novel tag update technique for dual system encryption, and then we applied this technique for our PKBE schemes to improve the efficiency of schemes.

One interesting open problem is to construct an adaptively secure PKBE scheme under standard assumptions with *constant size* of private keys. Note that our PKBE schemes and the AugBE schemes only provide sub-linear size of private keys since private keys should be randomized. Previously, PKBE schemes with constant size of private keys were achieved by generating a private key deterministically and employing q-type assumption [7]. To devise a PKBE scheme with constant size of private keys under standard assumptions, we may need to invent a new technique.

References

[1] Dan Boneh and Xavier Boyen. Efficient selective-id secure identity-based encryption without random oracles. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT*, volume 3027 of *Lecture Notes in Computer Science*, pages 223–238. Springer, 2004.

[2] Dan Boneh and Xavier Boyen. Short signatures without random oracles. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT*, volume 3027 of *Lecture Notes in Computer Science*, pages 56–73. Springer, 2004.

[3] Dan Boneh and Xavier Boyen. Short signatures without random oracles and the sdh assumption in bilinear groups. *J. Cryptology*, 21(2):149–177, 2008.

[4] Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity based encryption with constant size ciphertext. In Ronald Cramer, editor, *EUROCRYPT*, volume 3494 of *Lecture Notes in Computer Science*, pages 440–456. Springer, 2005.

[5] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In Matthew K. Franklin, editor, *CRYPTO*, volume 3152 of *Lecture Notes in Computer Science*, pages 41–55. Springer, 2004.

[6] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. In Joe Kilian, editor, *CRYPTO*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer, 2001.

[7] Dan Boneh, Craig Gentry, and Brent Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In Victor Shoup, editor, *CRYPTO*, volume 3621 of *Lecture Notes in Computer Science*, pages 258–275. Springer, 2005.

[8] Dan Boneh, Amit Sahai, and Brent Waters. Fully collusion resistant traitor tracing with short ciphertexts and private keys. In Serge Vaudenay, editor, *EUROCRYPT*, volume 4004 of *Lecture Notes in Computer Science*, pages 573–592. Springer, 2006.

[9] Dan Boneh and Brent Waters. A fully collusion resistant broadcast, trace, and revoke system. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM Conference on Computer and Communications Security*, pages 211–220. ACM, 2006.

[10] Jung Hee Cheon. Security analysis of the strong diffie-hellman problem. In Serge Vaudenay, editor, *EUROCRYPT*, volume 4004 of *Lecture Notes in Computer Science*, pages 1–11. Springer, 2006.

[11] Benny Chor, Amos Fiat, and Moni Naor. Tracing traitors. In Yvo Desmedt, editor, *CRYPTO*, volume 839 of *Lecture Notes in Computer Science*, pages 257–270. Springer, 1994.

[12] Cécile Delerablée. Identity-based broadcast encryption with constant size ciphertexts and private keys. In Kaoru Kurosawa, editor, *ASIACRYPT*, volume 4833 of *Lecture Notes in Computer Science*, pages 200–215. Springer, 2007.

[13] Cécile Delerablée, Pascal Paillier, and David Pointcheval. Fully collusion secure dynamic broadcast encryption with constant-size ciphertexts or decryption keys. In Tsuyoshi Takagi, Tatsuaki Okamoto, Eiji Okamoto, and Takeshi Okamoto, editors, *Pairing*, volume 4575 of *Lecture Notes in Computer Science*, pages 39–59. Springer, 2007.

[14] Yevgeniy Dodis and Nelly Fazio. Public key broadcast encryption for stateless receivers. In Joan Feigenbaum, editor, *Digital Rights Management Workshop*, volume 2696 of *Lecture Notes in Computer Science*, pages 61–80. Springer, 2002.

[15] Amos Fiat and Moni Naor. Broadcast encryption. In Douglas R. Stinson, editor, *CRYPTO*, volume 773 of *Lecture Notes in Computer Science*, pages 480–491. Springer, 1993.

[16] Sanjam Garg, Abishek Kumarasubramanian, Amit Sahai, and Brent Waters. Building efficient fully collusion-resilient traitor tracing and revocation schemes. In Ehab Al-Shaer, Angelos D. Keromytis, and Vitaly Shmatikov, editors, *ACM Conference on Computer and Communications Security*, pages 121–130. ACM, 2010.

[17] Craig Gentry and Brent Waters. Adaptive security in broadcast encryption systems (with short ciphertexts). In Antoine Joux, editor, *EUROCRYPT*, volume 5479 of *Lecture Notes in Computer Science*, pages 171–188. Springer, 2009.

[18] Allison B. Lewko, Tatsuaki Okamoto, Amit Sahai, Katsuyuki Takashima, and Brent Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In Henri Gilbert, editor, *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 62–91. Springer, 2010.

[19] Allison B. Lewko, Amit Sahai, and Brent Waters. Revocation systems with very small private keys. In *IEEE Symposium on Security and Privacy*, pages 273–285. IEEE Computer Society, 2010.

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

[21] Allison B. Lewko and Brent Waters. Unbounded hibe and attribute-based encryption. In Kenneth G. Paterson, editor, *EUROCRYPT*, volume 6632 of *Lecture Notes in Computer Science*, pages 547–567. Springer, 2011.

[22] Ben Lynn. The pairing-based cryptography library. `http://crypto.stanford.edu/pbc/`.

[23] Dalit Naor, Moni Naor, and Jeffery Lotspiech. Revocation and tracing schemes for stateless receivers. In Joe Kilian, editor, *CRYPTO*, volume 2139 of *Lecture Notes in Computer Science*, pages 41–62. Springer, 2001.

[24] Moni Naor and Benny Pinkas. Efficient trace and revoke schemes. In Yair Frankel, editor, *Financial Cryptography*, volume 1962 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2000.

[25] Tatsuaki Okamoto and Katsuyuki Takashima. Adaptively attribute-hiding (hierarchical) inner product encryption. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT*, volume 7237 of *Lecture Notes in Computer Science*, pages 591–608. Springer, 2012.

[26] Jong Hwan Park, Hee Jean Kim, H.-M. Sung, and Dong Hoon Lee. Public key broadcast encryption schemes with shorter transmissions. *IEEE Trans. Broadcast.*, 54(3):401–411, 2008.

[27] Jong Hwan Park, Hyun Sook Rhee, and Dong Hoon Lee. Fully collusion-resistant trace-and-revoke scheme in prime-order groups. *Journal of Communications and Networks*, 13(5):428–441, 2011.

[28] Brent Waters. Dual system encryption: Realizing fully secure ibe and hibe under simple assumptions. In Shai Halevi, editor, *CRYPTO*, volume 5677 of *Lecture Notes in Computer Science*, pages 619–636. Springer, 2009.

[29] Brent Waters. Dual system encryption: Realizing fully secure ibe and hibe under simple assumptions. Cryptology ePrint Archive, Report 2009/385, 2009. `http://eprint.iacr.org/2009/385`.

[30] Brent Waters. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In Dario Catalano, Nelly Fazio, Rosario Gennaro, and Antonio Nicolosi, editors, *Public Key Cryptography*, volume 6571 of *Lecture Notes in Computer Science*, pages 53–70. Springer, 2011.