Adaptively Secure Distributed Broadcast Encryption with Linear-Size Public Parameters

Kwangsu Lee*

Abstract

Distributed broadcast encryption (DBE) is a variant of broadcast encryption (BE) that can efficiently transmit a message to a subset of users, in which users independently generate user private keys and user public keys instead of a central trusted authority generating user keys. In this paper, we propose a DBE scheme with constant size ciphertexts, constant size private keys, and linear size public parameters, and prove the adaptive security of our DBE scheme under static assumptions in composite-order bilinear groups. The previous efficient DBE schemes with constant size ciphertexts and constant size private keys are proven secure under the q-Type assumption or have a drawback of having quadratic size public parameters. In contrast, our DBE scheme is the first DBE scheme with linear size public parameters proven adaptively secure under static assumptions in composite-order bilinear groups.

Keywords: Broadcast encryption, Distributed broadcast encryption, Adaptive security, Bilinear maps.

*Sejong University, Seoul, Korea. Email: kwangsu@sejong.ac.kr.

1 Introduction

Broadcast encryption (BE) is a special kind of an encryption mechanism in which a ciphertext is associated with a set of recipients, and a user belonging to the set of recipients can decrypt the ciphertext with their own private key [9]. A non-trivial BE scheme must have sublinear size ciphertexts since a trivial BE scheme with linear size ciphertexts can be easily constructed by simply concatenating ciphertexts of public-key encryption (PKE). Many public-key BE schemes with constant size ciphertexts that allow anyone to create a ciphertext have been proposed [4, 15, 21]. However, the biggest drawback of existing efficient BE schemes is that a central trusted authority is required to generate private keys of users. In PKE, a central trusted authority is not needed for key generation because individual users independently generate private keys and public keys. The need for a central trusted authority is an obstacle that hinders the application of BE schemes to decentralized environments that have recently been attracting attention, such as blockchains.

Distributed broadcast encryption (DBE) is a variant of BE in which users can independently generate their own private and public keys, and there is no need of a central trusted authority for key generation [5,23]. As a result, the encryption and decryption algorithms of DBE require the public keys of recipients in addition to public parameters, which increases the storage size for storing the public keys of users. Previously, many DBE schemes have been proposed by using bilinear pairing, indistinguishability obfuscation, and lattices [5, 7, 18, 23]. In reality, the most efficient DBE schemes are those designed in pairing groups, which have $O(1)$ ciphertext size, $O(1)$ user private key size, $O(L)$ user public key size, and $O(L)$ or $O(L^2)$ public parameters size where L is the number of users [18]. In addition, these efficient DBE schemes provide adaptive security that allows an attacker to select the target set for attacks in the challenge phase.

The most efficient DBE scheme is the KMW-DBE scheme with linear size public parameters which is derived from the BGW-BE scheme in prime-order bilinear groups, which has been proven to be adaptive secure under the q-Type assumption [18]. In general, the q-Type assumption has been widely used to prove the security of efficient BE schemes, but it has a disadvantage that the security decreases as the parameter q increases where q is dependent on the number of private keys. In this paper, we ask whether it is possible to construct a DBE scheme with linear size public parameters and prove the adaptive security under static assumptions instead of the q-Type assumption.

1.1 Our Contributions

In this paper, we first propose an efficient DBE scheme with linear-size public parameters and prove its semi-static security based on static assumptions in composite-order bilinear groups. The semi-static security model which was introduced by Gentry and Waters [15] is weaker than the adaptive security model where an attacker selects the target set in the challenge phase, but stronger than the static security model where the attacker must submit the target set in the initial phase [4]. However, a semi-statically secure DBE scheme can be converted into an adaptively secure DBE scheme by using the conversion method proposed by Gentry and Waters [15, 18]. Thus, we obtain the first DBE scheme that has constant size ciphertexts, constant size user private keys, linear size user public key, and linear-size public parameters, and prove the adaptive security of our DBE scheme under static assumptions in composite-order bilinear groups. The comparison of our DBE scheme with the previous BE and DBE schemes is given in Table 1.

The basic idea of designing a DBE scheme is to decentralize the private key generation process of a BE scheme [18]. The most efficient BE scheme is the BGW-BE scheme because it has linear size public parameters, constant size private keys, and constant size ciphertexts, but the static security of this scheme can be only proven under the q-Type assumption [4]. The KMW-DBE scheme is an efficient DBE scheme derived from the BGW-BE scheme by decentralizing the private key generation process, but its semi-static

Table 1: Comparison of broadcast encryption schemes in bilinear groups

Scheme	Type	PP	USK	UPK	CT	Model	Assumption
BGW [4]	BE	$O(L)$	$O(1)$	-	$O(1)$	ST	q-Type
AKN [1]	BE	$O(L)$	$O(L)$	-	$O(1)$	ST	q-Type
GW [15]	BE	$O(L)$	$O(L)$	-	$O(1)$	AD	q-Type
GW [15]	BE	$O(L)$	$O(1)$	-	$O(1)$	AD	q-Type
Waters [21]	BE	$O(L)$	$O(L)$	-	$O(1)$	AD	DLIN
Wee [22]	BE	$O(L)$	$O(1)$	-	$O(1)$	ST	SD, GSD
GKW [14]	BE	$O(L^2)$	$O(1)$	-	$O(1)$	AD	SD, GSD
GKW [14]	BE	$O(L^2)$	$O(1)$	-	$O(1)$	AD	k-LIN
HWW [17]	BE	$O(L)$	$O(1)$	-	$O(1)$	AD	q-Type
WQZD [23]	DBE	$O(L)$	$O(L)$	$O(L^2)$	$O(1)$	AD	q-Type
KMW [18]	DBE	$O(L)$	$O(1)$	$O(L)$	$O(1)$	AD	q-Type
KMW [18]	DBE	$O(L^2)$	$O(1)$	$O(L)$	$O(1)$	AD	k-LIN
Ours	DBE	$O(L)$	$O(1)$	$O(L)$	$O(1)$	AD	SD, GSD

Let L be the number of all users. We count the number of group elements to measure the size. We use symbols ST for static security and AD for adaptive security.

security can be only proven under the q-Type assumption by using the partitioning technique [18]. Our DBE scheme is a modification of the KMW-DBE scheme to use composite-order bilinear groups instead of prime-order bilinear groups. To prove the semi-static security of our DBE scheme under static assumptions instead of the q-Type assumption, we use the dual system encryption technique and its variant, which were widely used in the proofs of existing identity-based encryption (IBE), hierarchical IBE (HIBE), and attribute-based encryption (ABE) schemes [8, 19, 21, 22].

1.2 Related Work

Broadcast Encryption. The concept of broadcast encryption (BE), which can securely transmit a message to a subset of users, was introduced by Fiat and Naor [9]. Naor et al. proposed symmetric-key BE schemes by using the subset cover framework and showed that their schemes provide collusion resistance security [20]. The ciphertexts of symmetric-key BE can only be created by a central trusted authority, but the ciphertexts of public-key BE can be created by anyone. Boneh et al. proposed the first public-key BE scheme which has constant size ciphertexts and constant size private keys in prime-order bilinear groups and proved its static security under the q-Type assumption [4]. Abdalla et al. showed that it is possible to convert an HIBE scheme with the private key delegation property into a BE scheme and proposed an efficient BE scheme with constant size ciphertexts and linear size private keys based on the BBG-HIBE scheme [1]. The ideal security model of BE is an adaptive security model in which an attacker selects the target subsets in the challenge phase. Gentry and Waters presented a conversion method that transform a semi-statically secure BE scheme into an adaptively secure BE scheme and proposed an adaptively secure

identity-based BE scheme [15]. Waters proposed a BE scheme with constant size private keys and proved its adaptive security by using the dual system encryption technique under the standard assumption [21]. Gay et al. presented an efficient BE scheme with square size public parameters, constant size ciphertexts, and constant size private keys, and proved its adaptive security under standard assumptions [14]. Hsieh et al. presented another BE scheme which reduces the square size public parameters of the GKW-BE scheme to linear size public parameters by compressing the public parameters [17].

Distributed Broadcast Encryption. While BE requires a central trusted authority that generates users' private keys, distributed BE (DBE) does not require the central trusted authority since it allows users to independently generate their own private and public keys [5, 23]. Wu et al. proposed the concept of Ad hoc broadcast encryption (AHBE) that does not require a central trusted authority and proposed an AHBE scheme with relatively large size private and public keys [23]. Boneh and Zhandry proposed the concept of DBE and showed that the most efficient DBE scheme can be constructed by using indistinguishability obfuscation [5]. Kolonelos et al. showed that it is possible to convert existing BGW-BE and GKW-BE schemes in bilinear groups into DBE schemes and proposed an efficient DBE scheme that provides the adaptive security using DSE technique under the standard assumption [18]. The GW transformation that converts a semi-statically secure BE scheme into an adaptive secure BE scheme is equally applicable to DBE schemes [18]. Recently, Champion and Wu proposed the first DBE scheme based on lattices and proved its security under the modified LWE assumption [7]. Garg et al. introduced the concept of flexible broadcast encryption (FBE) that does not require to specify a user index when generating a user secret key and proposed a conversion method to convert a DBE scheme into an FBE scheme [10]. We may view DBE is a special case of silent threshold encryption (STE) that supports distributed private key generation, and Garg et al. proposed a secure and efficient STE scheme in the generic group model and showed that an efficient DBE scheme can be derived from their STE scheme [13].

Registration-Based Encryption. Identity-based encryption (IBE) is a variant of public-key encryption in which the public key of a user is replaced by an identity string, and it requires a trusted authority to generate a private key corresponding to the identity of a user [3]. Registration-based encryption (RBE) is an extension of IBE that replaces the trusted authority with a key curator who simply registers public keys of users without knowledge of any secret keys [12]. Recently, the concept of registered attribute-based encryption (Reg-ABE) was also introduced by applying RBE to attribute-based encryption (ABE), and an efficient Reg-ABE scheme in bilinear groups was proposed [16]. Since ABE can play the role of BE, a Reg-ABE scheme can be naturally converted to a DBE scheme. Many Reg-ABE schemes have been proposed in bilinear groups and lattices [2, 6, 11, 24].

2 Preliminaries

In this section, we define symmetric-key encryption, the bilinear groups of composite-order, and complexity assumptions.

2.1 Symmetric Key Encryption

Definition 2.1 (Symmetric Key Encryption)**.** A symmetric key encryption (SKE) scheme consists of three algorithms **GenKey**, **Encrypt**, and **Decrypt**, which are defined as follows:

GenKey(1^λ): The key generation algorithm takes as input a security parameter λ. It outputs a symmetric key K.

Encrypt(K,M): The encryption algorithm takes as input a symmetric key K and a message M. It outputs a ciphertext C.

Decrypt(K,C): The decryption algorithm takes as input a symmetric key K and a ciphertext C. It outputs a message M or a special symbol $\perp$.

The correctness property of SKE is defined as follows: For all K generated by **GenKey**(1^λ) and any message M, it is required that **Decrypt**$(K,$**Encrypt**$(K,M)) = M$.

Definition 2.2 (One-Message Indistinguishability)**.** The one-message indistinguishability (OMI) of SKE is defined in terms of the following experiment between a challenger $\mathcal{C}$ and a PPT adversary $\mathcal{A}$ where 1^λ is given as input:

1. **Setup**: $\mathcal{C}$ obtains a symmetric key K by running **GenKey**(1^λ) and keeps K to itself.
2. **Challenge**: $\mathcal{A}$ submits challenge messages M_0^*, M_1^* where $|M_0^*| = |M_1^*|$. $\mathcal{C}$ flips a random coin $\mu \in \{0,1\}$ and obtains CT^* by running **Encrypt**(K, M_μ^*). It gives CT^* to $\mathcal{A}$.
3. **Guess**: $\mathcal{A}$ outputs a guess $\mu' \in \{0,1\}$. $\mathcal{C}$ outputs 1 if $\mu = \mu'$ or 0 otherwise.

The advantage of $\mathcal{A}$ is defined as $\mathbf{Adv}^{OMI}_{SKE,\mathcal{A}}(\lambda) = \left|\Pr[\mu = \mu'] - \frac{1}{2}\right|$ where the probability is taken over all the randomness of the experiment. An SKE scheme is OMI secure if for all probabilistic polynomial-time (PPT) adversary $\mathcal{A}$, the advantage of $\mathcal{A}$ is negligible in the security parameter λ.

2.2 Bilinear Groups of Composite Order

Let $N = p_1 p_2 p_3$ where p_1, p_2, and p_3 are distinct prime numbers. Let $\mathbb{G}$ and $\mathbb{G}_T$ be two multiplicative cyclic groups of same composite order N and g be a generator of $\mathbb{G}$. The bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ has the following properties:

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

We say that $\mathbb{G}$ is a bilinear group if the group operations in $\mathbb{G}$ and $\mathbb{G}_T$ as well as the bilinear map e are all efficiently computable. Furthermore, we assume that the description of $\mathbb{G}$ and $\mathbb{G}_T$ includes generators of $\mathbb{G}$ and $\mathbb{G}_T$ respectively. We use the notation $\mathbb{G}_{p_i}$ to denote the subgroups of order p_i of $\mathbb{G}$ respectively. Similarly, we use the notation $\mathbb{G}_{T,p_i}$ to denote the subgroups of order p_i of $\mathbb{G}_T$ respectively. We note that if $h_i \in \mathbb{G}_{p_i}$ and $h_j \in \mathbb{G}_{p_j}$ for $i \neq j$, then $e(h_i, h_j)$ is the identity element in $\mathbb{G}_T$. This orthogonality property of $\mathbb{G}_{p_1}, \mathbb{G}_{p_2}, \mathbb{G}_{p_3}$ will be used to implement semi-functionality in our constructions.

2.3 Complexity Assumptions

Assumption 1 (Subgroup Decision, SD)**.** Let $(N, \mathbb{G}, \mathbb{G}_T, e)$ be a description of the bilinear group of composite order $N = p_1 p_2 p_3$. Let g_1, g_2, g_3 be generators of subgroups $\mathbb{G}_{p_1}, \mathbb{G}_{p_2}, \mathbb{G}_{p_3}$ respectively. The SD assumption is that if the challenge tuple

$$D = ((N, \mathbb{G}, \mathbb{G}_T, e), g_1, g_3) \text{ and } Z$$

are given, no PPT algorithm $\mathcal{A}$ can distinguish $Z = Z_0 = X_1 \in \mathbb{G}_{p_1}$ from $Z = Z_1 = X_1 R_1 \in \mathbb{G}_{p_1 p_2}$ with more than a negligible advantage. The advantage of $\mathcal{A}$ is defined as $\mathbf{Adv}^{SD}_{\mathcal{A}}(\lambda) = \left|\Pr[\mathcal{A}(D, Z_0) = 0] - \Pr[\mathcal{A}(D, Z_1) = 0]\right|$ where the probability is taken over random choices of $X_1 \in \mathbb{G}_{p_1}$ and $R_1 \in \mathbb{G}_{p_2}$.

Assumption 2 (General Subgroup Decision, GSD)**.** Let $(N, \mathbb{G}, \mathbb{G}_T, e)$ be a description of the bilinear group of composite order $N = p_1 p_2 p_3$. Let g_1, g_2, g_3 be generators of subgroups $\mathbb{G}_{p_1}, \mathbb{G}_{p_2}, \mathbb{G}_{p_3}$ respectively. The GSD assumption is that if the challenge tuple

$$D = ((N, \mathbb{G}, \mathbb{G}_T, e), g_1, g_3, X_1 R_1, R_2 Y_1) \text{ and } Z$$

are given, no PPT algorithm $\mathcal{A}$ can distinguish $Z = Z_0 = X_2 Y_2 \in \mathbb{G}_{p_1 p_3}$ from $Z = Z_1 = X_2 R_3 Y_2 \in \mathbb{G}_{p_1 p_2 p_3}$ with more than a negligible advantage. The advantage of $\mathcal{B}$ is defined as $\mathbf{Adv}^{GSD}_{\mathcal{A}}(\lambda) = \left| \Pr[\mathcal{A}(D, Z_0) = 0] - \Pr[\mathcal{A}(D, Z_1) = 0] \right|$ where the probability is taken over random choices of $X_1, X_2 \in \mathbb{G}_{p_1}$, $R_1, R_2, R_3 \in \mathbb{G}_{p_2}$, and $Y_1, Y_2 \in \mathbb{G}_{p_3}$.

3 Distributed Broadcast Encryption

In this section, we define the syntax of DBE and its security models.

3.1 Definition

In a DBE scheme, a trusted authority generates public parameters to be used in the system by running the setup algorithm. Each user generates a user private key and a user public key for the user's index by running the key generation algorithm with the public parameters as input and stores the user public key in a public directory. Then, a sender creates a ciphertext for a subset of users by running the encryption algorithm with the receivers' public keys and public parameters as input. A receiver can decrypt the ciphertext using his private key if its index belongs to the subset of the ciphertext. A more detailed syntax of the DBE scheme is given as follows.

Definition 3.1 (Distributed Broadcast Encryption)**.** A distributed broadcast encryption (DBE) scheme consists of five algorithms **Setup**, **GenKey**, **IsValid**, **Encaps**, and **Decaps**, which are defined as follows:

Setup$(1^\lambda, 1^L)$: The setup algorithm takes as input a security parameter 1^λ, and the number users L. It outputs public parameters PP.

GenKey(i, PP): The key generation algorithm takes as input a user index $i \in [L]$ and public parameters PP. It outputs a private key USK_i and a public key UPK_i.

IsValid(j, UPK_j, PP): The public key verification algorithm takes as input an index j, a public key UPK_j, and the public parameters PP. It outputs 1 or 0 depending on the validity of keys.

Encaps$(S, \{(j, UPK_j)\}_{j \in S}, PP)$: The encapsulation algorithm takes as input a set $S \subseteq [L]$, public keys $\{(j, UPK_j)\}_{j \in S}$, and public parameters PP. It outputs a ciphertext header CH and a session key CK.

Decaps$(S, CH, i, USK_i, \{(j, UPK_j)\}_{j \in S}, PP)$: The decapsulation algorithm takes as input a set S, a ciphertext header CH, an index i, a private key USK_i for the index i, public keys $\{(j, UPK_j)\}_{j \in S}$, and public parameters PP. It outputs a session key CK or $\perp$.

The correctness of DBE is defined as follows: For all PP generated by **Setup**$(1^\lambda, 1^L)$, all (USK_i, UPK_i) generated by **GenKey**(i, PP), all UPK_j such that **IsValid**(j, UPK_j, PP), all $S \subseteq [L]$, it is required that

- If $i \in S$, then $CK = CK'$ where $(CH, CK) =$ **Encaps**$(S, \{(j, UPK_j)\}_{j \in S}, PP)$ and $CK' =$ **Decaps**$(S, CH, i, USK_i, \{(j, UPK_j)\}_{j \in S}, PP)$.

3.2 Security Model

The semi-static security model is an enhanced security model of the static security model in which an attacker specifies the challenge set S^* before it sees the public parameters [15]. In the semi-static security model, an attacker first commits an initial set $\tilde{S}$, and a challenger generates public parameter PP and gives it to the attacker. Afterwards, the attacker can obtain the public keys of users belonging to $\tilde{S}$. In the challenge phase, the attacker submits the challenge set S^*, which is a subset of $\tilde{S}$, and obtains the challenge ciphertext header CH^* and the challenge session key CK^*_μ. Finally, the attacker succeeds if he can guess whether the challenge session key is correct or random. The detailed description of this security model is given as follows:

Definition 3.2 (Semi-Static Security)**.** The semi-static security of DBE is defined in terms of the following experiment between a challenger $\mathcal{C}$ and a PPT adversary $\mathcal{A}$ where 1^λ and 1^L are given as input:

1. **Init**: $\mathcal{A}$ initially commits an initial set $\tilde{S} \subseteq [L]$.
2. **Setup**: $\mathcal{C}$ obtains public parameters PP by running **Setup**$(1^\lambda, 1^L)$ and gives PP to $\mathcal{A}$.
3. **Query Phase**: $\mathcal{C}$ generates a key pair (USK_j, UPK_j) by running **GenKey**(j, PP) for all $j \in \tilde{S}$. It gives $\{(j, UPK_j)\}_{j \in \tilde{S}}$ to $\mathcal{A}$.
4. **Challenge**: $\mathcal{A}$ submits a challenge set $S^* \subseteq \tilde{S}$. $\mathcal{C}$ obtains a ciphertext tuple (CH^*, CK^*) by running **Encaps**$(S^*, \{(j, UPK_j)\}_{j \in S^*}, PP)$. It sets $CK^*_0 = CK^*$ and $CK^*_1 = RK$ by selecting a random RK. It flips a random coin $\mu \in \{0,1\}$ and gives (CH^*, CK^*_μ) to $\mathcal{A}$.
5. **Guess**: Finally, $\mathcal{A}$ outputs a guess $\mu' \in \{0,1\}$, and wins the game if $\mu = \mu'$.

The advantage of $\mathcal{A}$ is defined as $\mathbf{Adv}^{SS}_{DBE,\mathcal{A}}(\lambda) = \left| \Pr[\mu = \mu'] - \frac{1}{2} \right|$ where the probability is taken over all the randomness of the experiment. A DBE scheme is semi-statically secure if for all probabilistic polynomial-time (PPT) adversary $\mathcal{A}$, the advantage of $\mathcal{A}$ is negligible in the security parameter λ.

The adaptive security model is the strongest security model of BE in which an attacker can specify the challenge set S^* in the challenge phase [15]. In the adaptive security model, a challenger first generates public parameter PP and gives it to an attacker. Then, the attacker requests a key generation query for a user index to obtain the user's public key, and a key reveal query for a user index to obtain the user's private key. In the challenge phase, the attacker submits the challenge set S^* that does not include the user index in key exposure queries and obtains the challenge ciphertext header CH^* and the challenge session key CK^*_μ. Finally, the attacker succeeds if he can guess whether the challenge session key is correct or random.

Definition 3.3 (Adaptive Security)**.** The adaptive security of DBE is defined in terms of the following experiment between a challenger $\mathcal{C}$ and a PPT adversary $\mathcal{A}$ where 1^λ and 1^L are given as input:

1. **Setup**: $\mathcal{C}$ obtains public parameters PP by running **Setup**$(1^\lambda, 1^L)$ and gives PP to $\mathcal{A}$.
2. **Query Phase**: $\mathcal{A}$ adaptively requests key generation and key corruption queries. These queries are processed as follows:
   - Key Generation: $\mathcal{A}$ issues this query on an index $i \in [L]$ such that $i \notin KQ$. $\mathcal{C}$ creates (USK_i, UPK_i) by running **GenKey**(i, PP), adds i to KQ, and responds UPK_i to $\mathcal{A}$.
   - Key Corruption: $\mathcal{A}$ issues this query on an index $i \in [L]$ such that $i \in KQ \setminus CQ$. $\mathcal{C}$ adds i to CQ and responds USK_i to $\mathcal{A}$.

3. **Challenge**: $\mathcal{A}$ submits a challenge set $S^* \subseteq KQ \setminus CQ$. $\mathcal{C}$ obtains a ciphertext tuple (CH^*, CK^*) by running $\mathbf{Encaps}(S^*, \{(j, UPK_j)\}_{j \in S^*}, PP)$. It sets $CK_0^* = CK^*$ and $CK_1^* = RK$ by selecting a random RK. It flips a random coin $\mu \in \{0,1\}$ and gives (CH^*, CK_μ^*) to $\mathcal{A}$.

4. **Guess**: Finally, $\mathcal{A}$ outputs a guess $\mu' \in \{0,1\}$, and wins the game if $\mu = \mu'$.

The advantage of $\mathcal{A}$ is defined as $\mathbf{Adv}_{DBE,\mathcal{A}}^{AD}(\lambda) = \left| \Pr[\mu = \mu'] - \frac{1}{2} \right|$ where the probability is taken over all the randomness of the experiment. A DBE scheme is adaptively secure if for all probabilistic polynomial-time (PPT) adversary $\mathcal{A}$, the advantage of $\mathcal{A}$ is negligible in the security parameter λ.

Lemma 3.1 ([15, 18]). *Let Π_{SS} be a semi-statically secure DBE scheme. Then there exists Π_{AD} that is an adaptively secure DBE scheme.*

The active-adaptive security model is a modification of the adaptive security model for DBE to allow the registration of malicious user public keys [18]. This active-adaptive security model is very similar to the adaptive security model above except that an attacker additionally requests a malicious corruption query to register a malicious user public key. The detailed description of this security model is given as follows:

Definition 3.4 (Active-Adaptive Security)**.** The active-adaptive security of DBE is defined in terms of the following experiment between a challenger $\mathcal{C}$ and a PPT adversary $\mathcal{A}$ where 1^λ and 1^L are given as input:

1. **Setup**: $\mathcal{C}$ obtains public parameters PP by running $\mathbf{Setup}(1^\lambda, 1^L)$ and gives PP to $\mathcal{A}$.

2. **Query Phase**: $\mathcal{A}$ adaptively requests key generation, key corruption, and malicious corruption queries. These queries are processed as follows:

   - Key Generation: $\mathcal{A}$ issues this query on an index $i \in [L]$ such that $i \notin KQ \wedge i \notin MQ$. $\mathcal{C}$ creates (USK_i, UPK_i) by running $\mathbf{GenKey}(i, PP)$, adds i to KQ, and responds UPK_i to $\mathcal{A}$.
   - Key Corruption: $\mathcal{A}$ issues this query on an index $i \in [L]$ such that $i \in KQ \wedge i \notin CQ$. $\mathcal{C}$ adds i to CQ and responds with USK_i to $\mathcal{A}$.
   - Malicious Corruption: $\mathcal{A}$ issues this query on an index $i \in [L]$ such that $i \notin KQ \wedge i \notin MQ$. $\mathcal{C}$ adds i to MQ and stores UPK_i.

3. **Challenge**: $\mathcal{A}$ submits a challenge set $S^* \subseteq KQ \setminus (CQ \cup MQ)$. $\mathcal{C}$ obtains a ciphertext tuple (CH^*, CK^*) by running $\mathbf{Encaps}(S^*, \{(j, UPK_j)\}_{j \in S^*}, PP)$. It sets $CK_0^* = CK^*$ and $CK_1^* = RK$ by selecting a random RK. It flips a random coin $\mu \in \{0,1\}$ and gives (CH^*, CK_μ^*) to $\mathcal{A}$.

4. **Guess**: Finally, $\mathcal{A}$ outputs a guess $\mu' \in \{0,1\}$, and wins the game if $\mu = \mu'$.

The advantage of $\mathcal{A}$ is defined as $\mathbf{Adv}_{DBE,\mathcal{A}}^{AA}(\lambda) = \left| \Pr[\mu = \mu'] - \frac{1}{2} \right|$ where the probability is taken over all the randomness of the experiment. A DBE scheme is active-adaptively secure if for all probabilistic polynomial-time (PPT) adversary $\mathcal{A}$, the advantage of $\mathcal{A}$ is negligible in the security parameter λ.

Lemma 3.2 ([18]). *Let Π_{AD} be an adaptively secure DBE scheme. Then Π_{AD} is also active-adaptively secure.*

4 Construction

In this section, we propose a basic DBE scheme for the semi-static security and an enhanced DBE scheme for the adaptive security.

4.1 Semi-Static Construction

Our basic DBE_{SS} scheme has a similar structure to the KMW-DBE scheme with linear size public parameters, constant size ciphertexts, and constant size private keys of Kolonelos et al. [18]. However, our DBE_{SS} scheme uses composite-order bilinear groups instead of prime-order bilinear groups and modifies some group elements to use the dual system encryption technique in the security proof. The detailed description of our basic DBE_{SS} scheme is given as follows:

DBE_{SS}.Setup$(1^\lambda, 1^L)$**:** Let λ be a security parameter and L be the maximum number of users. It first generates bilinear groups $\mathbb{G}, \mathbb{G}_T$ of composite order $N = p_1 p_2 p_3$ where p_1, p_2, and p_3 are random primes. It selects random generators g_1, g_3 of $\mathbb{G}_{p_1}, \mathbb{G}_{p_3}$ respectively. It selects random $\alpha \in \mathbb{Z}_N$ and $u \in \mathbb{G}_{p_1}$. Next, it selects random $\{Y_k\}_{1 \le k \le 2L} \in \mathbb{G}_{p_3}$ and creates $\{A_k = g^{\alpha^k}\}_{1 \le k \le L}$, $\{U_k = u^{\alpha^k} Y_k\}_{1 \le k \le 2L}$. It chooses a pairwise independent hash function $\mathbf{H}$ such that $\mathbf{H} : \mathbb{G}_T \to \{0,1\}^\lambda$. It outputs public parameters

$$PP = \Big((N, \mathbb{G}, \mathbb{G}_T, e),\ g = g_1, Y = g_3,\ \{A_k\}_{1 \le k \le L},\ \{U_k\}_{1 \le k \ne L+1 \le 2L},\ \Omega = e(g, U_{L+1}),\ \mathbf{H} \Big).$$

DBE_{SS}.GenKey(i, PP)**:** It selects random $\gamma_i \in \mathbb{Z}_N$ and $\{Y_k\}_{1 \le k \le L} \in \mathbb{G}_{p_3}$. It outputs a private key USK_i and a public key UPK_i as

$$USK_i = \Big(K_i = U_{L+1-i}^{\gamma_i} Y_{L+1-i} \Big),\ UPK_i = \Big(V_i = g^{\gamma_i},\ \{V_{i,k} = U_k^{\gamma_i} Y_k\}_{1 \le k \ne L+1-i \le L} \Big).$$

DBE_{SS}.IsValid(j, UPK_j, PP)**:** Let $UPK_j = (V_j, \{V_{j,k}\})$. It computes $T = e(V_j, U_L)$. For all $k \in \{1, \ldots, L\} \setminus \{L+1-j\}$, it checks that $T \stackrel{?}{=} e(A_{L-k}, V_{j,k})$ where $A_0 = g$. If it passes all checks, then it outputs 1. Otherwise, it outputs 0.

DBE_{SS}.Encaps$(S, \{(j, UPK_j)\}_{j \in S}, PP)$**:** Let $UPK_j = (V_j, \{V_{j,k}\})$. It selects random $t \in \mathbb{Z}_N$ and outputs a ciphertext header

$$CH = \Big(C_1 = g^t,\ C_2 = \Big(\prod_{j \in S} A_j V_j \Big)^t \Big)$$

and a session key $CK = \mathbf{H}(\Omega^t)$.

DBE_{SS}.Decaps$(S, CH, i, USK_i, \{(j, UPK_j)\}_{j \in S}, PP)$**:** Let $CT = (C_1, C_2, C), USK_i = K_i$, and $UPK_j = (V_j, \{V_{j,k}\})$. If $i \notin S$, it outputs $\perp$. It computes decryption components

$$D_1 = K_i,\ D_2 = U_{L+1-i},\ D_3 = \prod_{j \in S \setminus \{i\}} U_{L+1-i+j} V_{j,L+1-i}.$$

It outputs a session key $CK = \mathbf{H}(e(C_2, D_2) \cdot e(C_1, D_1 \cdot D_3)^{-1})$.

To show the correctness of the basic DBE scheme, we show that a correct session key can be derived. If

$i \in S$, then we can check that a session element is derived by the following equation

$$
\begin{aligned}
e(C_2, D_2) &= e((\prod_{j\in S} A_j V_j)^t, U_{L+1-i}) \\
&= e((A_i V_i)^t, U_{L+1-i}) \cdot e((\prod_{j\in S\setminus\{i\}} A_j V_j)^t, U_{L+1-i}) \\
&= e((g^{\alpha^i})^t, u^{\alpha^{L+1-i}} Y) \cdot e((g^{\gamma_i})^t, u^{\alpha^{L+1-i}} Y) \cdot e((\prod_{j\in S\setminus\{i\}} g^{\alpha^j} g^{\gamma_j})^t, u^{\alpha^{L+1-i}} Y) \\
&= e(g^t, u^{\alpha^{L+1}}) \cdot e(g^t, u^{\alpha^{L+1-i}\gamma_i}) \cdot e(g^t, \prod_{j\in S\setminus\{i\}} u^{\alpha^{L+1-i+j}} \cdot u^{\alpha^{L+1-i}\gamma_j}) \\
&= \Omega^t \cdot e(C_1, K_i) \cdot e(C_1, \prod_{j\in S\setminus\{i\}} U_{L+1-i+j} V_{j,L+1-i}) = \Omega^t \cdot e(C_1, D_1 \cdot D_3).
\end{aligned}
$$

4.2 Adaptive Construction

Our enhanced DBE_{AD} scheme is derived by applying the transformation of Gentry and Waters [15] to our basic DBE_{SS} scheme. The GW transformation is a method that transforms a semi-statically secure BE scheme into an adaptively secure BE scheme and can be applied to DBE schemes as well. The detailed description of our DBE_{AD} scheme is given as follows:

DBE_{AD}.Setup$(1^\lambda, 1^L)$**:** Let λ be a security parameter and L be the number of users. It obtains PP_{SS} by running **DBE_{SS}.Setup**$(1^\lambda, 1^{2L})$. It outputs public parameters $PP = PP_{SS}$.

DBE_{AD}.GenKey(i, PP)**:** Let $i \in [L]$. It generates key pairs $(USK_{SS,2i}, UPK_{SS,2i})$ and $(USK_{SS,2i-1}, UPK_{SS,2i-1})$ by running **DBE_{SS}.GenKey**$(2i, PP_{SS})$ and **DBE_{SS}.GenKey**$(2i-1, PP_{SS})$ respectively. It selects a random bit $u \in \{0,1\}$ and erases $USK_{SS,2i-(1-u)}$ completely. It outputs a private key $USK_i = (USK_{SS,2i-u}, u)$ and a public key $UPK_i = (UPK_{SS,2i}, UPK_{SS,2i-1})$.

DBE_{AD}.IsValid(j, UPK_j, PP)**:** Let $UPK_j = (UPK_{SS,2j}, UPK_{SS,2j-1})$. It checks that **$\mathrm{DBE}_{SS}$.IsValid**$(2j, UPK_{SS,2j}, PP_{SS}) = 1$ and **DBE_{SS}.IsValid**$(2j-1, UPK_{SS,2j-1}, PP_{SS}) = 1$. If it passes all checks, then it outputs 1. Otherwise, it outputs 0.

DBE_{AD}.Encaps$(S, \{(j, UPK_j)\}_{j\in S}, PP)$**:** Let $S \subseteq [L]$ and $UPK_j = (UPK_{SS,2j}, UPK_{SS,2j-1})$.

1. It selects random bits $z = \{z_j\}_{j\in S}$ where $z_j \in \{0,1\}$. Next, it defines two sets $S_0 = \{2j - z_j\}_{j\in S}$ and $S_1 = \{2j - (1-z_j)\}_{j\in S}$.
2. It obtains two ciphertext pairs $(CH_{SS,0}, CK_{SS,0})$ and $(CH_{SS,1}, CK_{SS,1})$ by running **DBE_{SS}.Encaps** $(S_0, \{(k, UPK_{SS,k}\}_{k\in S_0}, PP_{SS})$ and **DBE_{SS}.Encaps**$(S_1, \{(k, UPK_{SS,k}\}_{k\in S_1}, PP_{SS})$ respectively.
3. It selects a random message $CK \in \{0,1\}^\lambda$. It obtains symmetric key ciphertexts CT_0 and CT_1 by running **SKE.Encrypt**$(CK_{SS,0}, CK)$ and **SKE.Encrypt**$(CK_{SS,1}, CK)$ respectively.
4. It outputs a ciphertext header $CH = (CH_{SS,0}, CH_{SS,1}, CT_0, CT_1, z)$ and a session key CK.

DBE_{AD}.Decaps$(S, CH, i, USK_i, \{(j, UPK_j)\}_{j\in S}, PP)$**:** Let $USK_i = (USK_{SS,2i-u}, u)$. If $i \notin S$, it outputs $\perp$.

1. It derives two sets $S_0 = \{2j - z_j\}_{j\in S}$ and $S_1 = \{2j - (1-z_j)\}_{j\in S}$ If $z_i = u$, then it sets $S' = S_0, CH'_{SS} = CH_{SS,0}, CT' = CT_0$. Otherwise, it sets $S' = S_1, CH'_{SS} = CH_{SS,1}, CT' = CT_1$.
2. It obtains CK'_{SS} by running **DBE_{SS}.Decaps**$(S', CH'_{SS}, 2i-u, USK_{SS,2i-u}, \{(k, UPK_{SS,k}\}_{k\in S'}, PP_{SS})$.

3. It obtains a decrypted message CK by running **SKE.Decrypt**(CK'_{SS}, CT') and outputs a session key CK.

The correctness of our enhanced DBE scheme easily followed from the correctness of the underlying SKE and DBE$_{SS}$ schemes.

5 Security Analysis

In this section, we show that our DBE$_{SS}$ scheme provides the semi-static security under static assumptions in composite-order bilinear groups. Then, we show that our DBE$_{AD}$ scheme provides the adaptive and active-adaptive security.

We prove the semi-static security of our DBE$_{SS}$ scheme by using the Déjà Q technique, which is a variant of the dual system encryption (DSE) technique [8, 21]. In particularly, we prove our DBE$_{SS}$ scheme by following the strategy of Wee [22] that was used to prove the static security of a variant BGW-BE scheme in composite-order bilinear groups. The basic idea of DSE proof is to change normal ciphertexts and normal private keys into semi-functional ciphertexts and semi-functional private keys through hybrid games. In the final game, since the semi-functional challenge ciphertext and semi-functional private keys are not related to each other, it is relatively easy for a simulator to generate semi-functional private keys that is not related to the challenge semi-functional ciphertext. Thus, it is possible to show that the challenge session key is random. The Déjà Q technique is very similar to the DSE technique except that it can be used to change normal private keys to semi-functional private keys even if the private keys do not have random variables.

Theorem 5.1 (Semi-Static Security)**.** *The basic DBE scheme is semi-statically secure if the SD and GSD assumptions hold.*

Proof. We first define the semi-functional type of elements and ciphertext. For the semi-functional type, we let g_2 denote a fixed generator of the subgroup $\mathbb{G}_{p_2}$.

UL-$(\eta, 0)$**.** Let $UL = \{U'_i = u^{\alpha^i} Y_i\}_{i=1}^{2L}$ be a normal list of elements. Let r_j, a_j be fixed random exponents for index $j \in [k]$. It selects random $Y'_1, \ldots, Y'_{2L} \in \mathbb{G}_{p_3}$ and outputs a type-$(\eta, 0)$ list of elements as

$$UL = \left\{ U_i = U'_i g_2^{\sum_{j=1}^{k-1} r_j a_j^{L+1-i}} Y'_i \right\}_{i=1}^{2L}.$$

UL-$(\eta, 1)$**.** Let $UL = \{U'_i = u^{\alpha^i} Y_i\}_{i=1}^{2L}$ be a normal list of elements. Let r_j, a_j be fixed random exponents for index $j \in [k]$. It selects random $Y'_1, \ldots, Y'_{2L} \in \mathbb{G}_{p_3}$ outputs a type-$(\eta, 1)$ list of elements as

$$UL = \left\{ U_i = U'_i g_2^{\sum_{j=1}^{k-1} r_j a_j^{L+1-i}} g_2^{r_k \alpha^{L+1-i}} Y'_i \right\}_{i=1}^{2L}.$$

UL-SF. Let $UL = \{U'_i = u^{\alpha^i} Y_i\}_{i=1}^{2L}$ be a list of normal elements. It chooses random $\delta_1, \ldots, \delta_{2L} \in \mathbb{Z}_N$, $Y'_1, \ldots, Y'_{2L} \in \mathbb{G}_{p_3}$ and outputs a semi-functional list of elements as

$$UL = \left\{ U_i = U'_i g_2^{\delta_i} Y'_i \right\}_{i=1}^{2L}.$$

CH-SF. Let $CH' = (C'_1, C'_2)$ be a normal ciphertext header. It chooses random $c, d \in \mathbb{Z}_N$ and outputs a semi-functional ciphertext header $CH = \left(C_1 = C'_1 g_2^c, C_2 = C'_2 g_2^{cd}\right)$.

The security proof consists of a sequence of hybrid games $\mathbf{G}_0, \mathbf{G}_1, \ldots, \mathbf{G}_5$. The first game will be the original semi-static security game and the last one will be a game in which an adversary has no advantage. We define the games as follows:

Game $\mathbf{G}_0$. This game is the original semi-static security game defined in Section 3.2. That is, the simulator of this game simply follows the honest algorithms. In this game, all parameters, key elements, and the challenge ciphertext are normal.

Game $\mathbf{G}_1$. This game is almost the same as the game $\mathbf{G}_0$ except that the simulator sets $\gamma_i = \gamma'_i - \alpha^i$ by selecting random $\gamma'_i \in \mathbb{Z}_N$ for each index i and creates the challenge session key $CK_0^* = \mathbf{H}(e(g^t, U_{L+1}))$ instead of $CK_0^* = \mathbf{H}(\Omega^t)$ where $U_{L+1} = u^{\alpha^{L+1}} Y_{L+1}$.

Game $\mathbf{G}_2$. In this game, the challenge ciphertext header is changed to be semi-functional, but all other elements are still normal.

Game $\mathbf{G}_3$. Next, we define a new game $\mathbf{G}_3$. In this game, we change the distribution of UL from normal to semi-functional. Because of this change, the public parameters, all key pairs, and the challenge session key that depend on UL also changed. For the analysis of this game, we define additional sub-games $\mathbf{H}_{1,0}, \mathbf{H}_{1,1}, \ldots, \mathbf{H}_{\eta,0}, \mathbf{H}_{\eta,1}, \ldots, \mathbf{H}_{2L,0}, \mathbf{H}_{2L,1}, \mathbf{H}_{2L+1,0}$ that change the type of elements in UL one by one where $\mathbf{H}_{1,0} = \mathbf{G}_2$ and $\mathbf{H}_{2L+1,0} = \mathbf{G}_3$. A more detailed definition of these sub-games is given as follows:

- **Game $\mathbf{H}_{\eta,0}$.** This game is similar to the game $\mathbf{G}_2$ excpet that the simulator generates a type-$(\eta, 0)$ list UL.
- **Game $\mathbf{H}_{\eta,1}$.** This game is also similar to the game $\mathbf{G}_2$ except the simulator generates a type-$(\eta, 1)$ list UL.

Game $\mathbf{G}_4$. In this game $\mathbf{G}_4$, the only change from the game $\mathbf{G}_3$ is that the simulator generates a semi-functional UL.

Game $\mathbf{G}_5$. In this final game $\mathbf{G}_5$, the challenge session key CK_0^* is changed to be random. Thus the adversary cannot distinguish the challenge session key.

Let $\mathbf{Adv}_{\mathcal{A}}^{G_j}$ be the advantage of $\mathcal{A}$ in the game $\mathbf{G}_j$. We have that $\mathbf{Adv}_{DBE,\mathcal{A}}^{SS}(\lambda) = \mathbf{Adv}_{\mathcal{A}}^{G_0}$, and $\mathbf{Adv}_{\mathcal{A}}^{G_5} = 0$. From the following Lemmas 5.2, 5.3, 5.4, 5.5, 5.6, and 5.7, we obtain the equation

$$\mathbf{Adv}_{DBE,\mathcal{A}}^{SS}(\lambda) \leq \sum_{j=1}^{5} \left| \mathbf{Adv}_{\mathcal{A}}^{G_{j-1}} - \mathbf{Adv}_{\mathcal{A}}^{G_j} \right| \leq \mathbf{Adv}_{\mathcal{B}}^{SD}(\lambda) + 2L \cdot \mathbf{Adv}_{\mathcal{B}}^{GSD}(\lambda) + O(L^2/p_2)$$

where L is the number of users. This completes the proof. □

Lemma 5.2. *No adversary can distinguish $\mathbf{G}_0$ from $\mathbf{G}_1$ since two games $\mathbf{G}_0$ and $\mathbf{G}_1$ are equal.*

Proof. Suppose there exists an adversary $\mathcal{A}$ that distinguishes $\mathbf{G}_0$ from $\mathbf{G}_1$ with a non-negligible advantage. $\mathcal{B}$ that interacts with $\mathcal{A}$ is described as follows:

Init: $\mathcal{A}$ submits an initial set $\tilde{S}$.

Setup: $\mathcal{B}$ chooses random $\alpha \in \mathbb{Z}_N$, $u \in \mathbb{G}_{p_1}$, $\{Y_k\}_{1 \leq k \leq 2L} \in \mathbb{G}_{p_3}$ and builds $\{A_k = g^{\alpha^k}\}_{1 \leq k \leq L}, UL = \{U_i = u^{\alpha^i} Y_i\}_{i=1}^{2L}$. It publishes

$$PP = \big((N, \mathbb{G}, \mathbb{G}_T, e), g = g_1, Y = g_3, \{A_k\}_{1 \leq k \leq L}, \{U_k\}_{1 \leq k \neq L+1 \leq 2L}, \Omega = e(g, U_{L+1})\big).$$

Query Phase: For each index $i \in \tilde{S}$, $\mathcal{B}$ selects random $\gamma'_i \in \mathbb{Z}_N$, $\{Y'_{i,k}\}_{1\le k\le L} \in \mathbb{G}_{p_3}$ and creates a public key

$$UPK_i = \left(V_i = g^{\gamma'_i} A_i^{-1}, \left\{V_{i,j} = U_j^{\gamma'_i} U_{j+i} Y'_{i,j}\right\}_{1\le j\neq L+1-i\le L}\right)$$

by implicitly setting $\gamma_i = \gamma'_i - \alpha^i$. It gives $\{(j, UPK_j)\}_{j\in\tilde{S}}$ to $\mathcal{A}$.

Challenge: For a challenge set $S^* \subseteq \tilde{S}$, $\mathcal{B}$ selects random $t \in \mathbb{Z}_N$ and creates a challenge ciphertext header and a session key

$$CH^* = \left(C_1^* = g^t,\ C_2^* = (g^t)^{\sum_{j\in S^*}\gamma'_j}\right),\ CK^* = \mathbf{H}(e(g^t, U_{L+1})).$$

It sets $CK_0^* = CK^*$ and $CK_1^* = RK$ by selecting a random RK. Next, it flips a random coin $\mu \in \{0,1\}$ and gives (CH^*, CK_μ^*) to $\mathcal{A}$

Guess: $\mathcal{A}$ outputs a guess μ'. If $\mu = \mu'$, then $\mathcal{B}$ outputs 1. Otherwise, it outputs 0.

This completes our proof. □

Lemma 5.3. *If the SD assumption holds, then no PPT adversary can distinguish $\mathbf{G}_1$ from $\mathbf{G}_2$ with a non-negligible advantage.*

Proof. Suppose there exists an adversary $\mathcal{A}$ that distinguishes $\mathbf{G}_0$ from $\mathbf{G}_1$ with a non-negligible advantage. A simulator $\mathcal{B}$ that solves the SD assumption using $\mathcal{A}$ is given: a challenge tuple $D = ((N, \mathbb{G}, \mathbb{G}_T, e), g_1, g_3)$ and Z where $Z = Z_0 = X_1 \in \mathbb{G}_{p_1}$ or $Z = Z_1 = X_1 R_1 \in \mathbb{G}_{p_1 p_2}$. The description of $\mathcal{B}$ that interacts with $\mathcal{A}$ is almost the same as that of Lemma 5.2 except the generation of the challenge ciphertext header. The challenge ciphertext header is generated as follows:

Challenge: For a challenge set S^*, $\mathcal{B}$ creates a challenge ciphertext header and a session key as

$$CH^* = \left(C_1^* = Z,\ C_2^* = (Z)^{\sum_{j\in S^*}\gamma'_j}\right),\ CK^* = \mathbf{H}(e(Z, U_{L+1}))$$

where $\{\gamma'_i\}$ are chosen in the key query step. It sets $CK_0^* = CK^*$ and $CK_1^* = RK$ by selecting a random RK. It flips a random coin $\mu \in \{0,1\}$ and gives (CH^*, CK_μ^*) to $\mathcal{A}$.

If $Z = Z_0 = X_1$, then the simulation is the same as $\mathbf{G}_0$. If $Z = Z_1 = X_1 R_1$, then it is the same as $\mathbf{G}_1$ since the challenge ciphertext header is semi-functional by implicitly setting $c \equiv \mathrm{dlog}(R_1) \bmod p_2, d \equiv \sum_{j\in S^*}\gamma'_j \bmod p_2$. Note that d is random since $\{\gamma'_j\}_{j\in S^*}$ modulo p_2 are not correlated with their values modulo p_1 by the Chinese Remainder Theorem (CRT). This completes our proof. □

Lemma 5.4. *If the GSD assumption holds, then no PPT adversary can distinguish $\mathbf{H}_{\eta,0}$ from $\mathbf{H}_{\eta,1}$ with a non-negligible advantage.*

Proof. Suppose there exists an adversary $\mathcal{A}$ that distinguishes $\mathbf{H}_{\eta,0}$ from $\mathbf{H}_{\eta,1}$ with a non-negligible advantage. A simulator $\mathcal{B}$ that solves the GSD assumption using $\mathcal{A}$ is given: a challenge tuple $D = ((N, \mathbb{G}, \mathbb{G}_T, e), g_1, g_3, X_1 R_1, R_2 Y_1)$ and Z where $Z = Z_0 = X_2 Y_2 \in \mathbb{G}_{p_1 p_3}$ or $Z = Z_1 = X_2 R_3 Y_2 \in \mathbb{G}_N$. Then $\mathcal{B}$ that interacts with $\mathcal{A}$ is almost similar to that of Lemma 5.3 except the generation of PP and the challenge ciphertext. The setup and challenge step is described as follows:

Setup: $\mathcal{B}$ chooses random $\alpha \in \mathbb{Z}_N$ and implicitly sets $u = X_2 \in \mathbb{G}_{p_1}$. It selects random $r_1, \ldots, r_{k-1}, a_1, \ldots, a_{k-1} \in \mathbb{Z}_N$ and builds a list of elements

$$UL = \left\{U_i = Z^{\alpha^i} (R_2 Y_1)^{\sum_{j=1}^{k-1} r_j a_j^i} Y'_i\right\}_{i=1}^{2L}$$

by selecting random $\{Y'_k\}_{1\le k\le 2L} \in \mathbb{G}_{p_3}$. It publishes $PP = \big((N,\mathbb{G},\mathbb{G}_T,e), g = g_1, Y = g_3, \{A_k = g^{\alpha^k}\}_{1\le k\le L}, \{U_k\}_{1\le k\ne L+1\le 2L}, \Omega = e(g,Z)^{\alpha^{L+1}}\big)$.

Challenge: For a challenge set S^*, $\mathcal{B}$ creates a challenge ciphertext header and a session key as

$$CH^* = \big(C_1 = X_1R_1,\ C_2 = (X_1R_1)^{\sum_{j\in S^*}\gamma'_j}\big),\ CK^* = \mathbf{H}(e(X_1R_1, U_{L+1})).$$

It sets $CK^*_0 = CK^*$ and $CK^*_1 = RK$ by selecting a random RK. It flips a random coin $\mu \in \{0,1\}$ and gives (CH^*, CK^*_μ) to $\mathcal{A}$.

Guess: $\mathcal{A}$ outputs a guess μ'. If $\mu = \mu'$, then $\mathcal{B}$ outputs 1. Otherwise, it outputs 0.

If $Z = Z_0 = X_2Y_2$, then the simulation is the same as $\mathbf{H}_{\eta,0}$. If $Z = Z_1 = X_2R_3Y_2$, then it is the same as $\mathbf{H}_{\eta,1}$ by implicitly setting $r_\eta \equiv \mathrm{dlog}(Z_1) \bmod p_2$. This completes our proof. □

Lemma 5.5. *No adversary can distinguish $\boldsymbol{H}_{\eta,1}$ from $\boldsymbol{H}_{k+1,0}$ since two games are equal.*

Proof. In the game $\mathbf{H}_{\eta,1}$ of the Lemma 5.4, the simulator builds an element U_i of UL. If we implicitly sets $X_2 = u, R_2 = g_2^{r''}, R_3 = g_2^{r_\eta}, r_j = r''r'_j$, and $a_\eta \equiv \alpha \bmod p_2$, then we can rewrite U_i as

$$\begin{aligned} U_i &= (X_2R_3Y_2)^{\alpha^i}(R_2Y_1)^{\sum_{j=1}^{\eta-1} r'_j a_j^i} Y_i' = X_2^{\alpha^i} R_2^{\sum_{j=1}^{\eta-1} r'_j a_j^i} R_3^{\alpha^i} Y' \\ &= u^{\alpha^i}(g_2^{r''})^{\sum_{j=1}^{\eta-1} r'_j a_j^i}(g_2^{r_\eta})^{a_\eta^i} Y' = u^{\alpha^i} g_2^{\sum_{j=1}^{\eta-1} r_j a_j^i} g_2^{r_\eta a_\eta^i} Y' \end{aligned}$$

since $a_\eta \equiv \alpha \bmod p_2$ is completely hidden by the CRT. This completes our proof. □

Lemma 5.6. *No adversary can distinguish $\boldsymbol{G}_3$ from $\boldsymbol{G}_4$ with a non-negligible advantage.*

Proof. In the game $\mathbf{G}_3$, the G_{p_2} parts of all elements in UL can be expressed as following matrix equation

$$\begin{pmatrix} a_1 & a_2 & \cdots & a_{2L} \\ a_1^2 & a_2^2 & \cdots & a_{2L}^2 \\ \vdots & \vdots & \ddots & \vdots \\ a_1^{2L} & a_2^{2L} & \cdots & a_{2L}^{2L} \end{pmatrix} \begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_{2L} \end{pmatrix} = \begin{pmatrix} \delta_1 \\ \delta_2 \\ \vdots \\ \delta_{2L} \end{pmatrix} \bmod p_2.$$

Since the left matrix is the Vandermonde matrix, this matrix is invertible if $a_1, \ldots, a_{2L}$ are distinct that can happen with probability $O(L^2/p_2)$. Thus there is one-to-one correspondence between $(r_1, \ldots, r_{2L})$ and $(\delta_1, \ldots, \delta_{2L})$. This completes the proof. □

Lemma 5.7. *No adversary can distinguish $\boldsymbol{G}_4$ from $\boldsymbol{G}_5$ with a non-negligible advantage.*

Proof. In the simulation of the game $\mathbf{G}_4$, the simulator generates all public keys only using $\{U_i\}_{1\le i\ne L+1\le 2L}$. That is, the $\mathbb{G}_{p_2}$ part of U_{L+1} is not revealed. Then the session key CK^*_0 is writted as

$$CK^*_0 = \mathbf{H}(e(C_1^*, U_{L+1})) = \mathbf{H}(e(C_1^*, u^{\alpha^{L+1}} g_2^{\delta_{L+1}})) = \mathbf{H}(e(C_1^*, u^{\alpha^{L+1}}) \cdot e(C_1^*, g_2^{\delta_{L+1}})).$$

Thus, CK^*_0 has additional $\log p_2$ bits of min-entropy from δ_{L+1} as long as δ_{L+1} is not zero. Then, by the leftover hash lemma, $\mathbf{H}(e(C_1^*, U_{L+1})$ is uniformly distributed since $\mathbf{H}$ is a pairwise independent hash function. □

Corollary 5.8 (Adaptive Security)**.** *The above DBE_{AD} scheme is adaptively secure if the DBE_{SS} scheme is semi-statically secure and the SKE scheme is OMI secure.*

The proof of this corollary is easily obtained from the Lemma 3.1.

Corollary 5.9 (Active-Adaptive Security)**.** *The above DBE_{AD} scheme is also active-adaptively secure if the DBE_{AD} scheme is adaptively secure.*

The proof of this corollary is also easily obtained from the Lemma 3.2.

6 Conclusion

In this paper, we proposed a DBE scheme with constant size ciphertexts, constant size private keys, and linear size public parameters, and proved the semi-static security under static assumptions in composite-order bilinear groups. We also showed that our DBE scheme can be converted an adaptively secure DBE scheme by doubling the ciphertext size using the GW transformation. Our DBE scheme is the first DBE scheme with linear size public parameters that is proven under static assumptions instead of the q-Type assumption. An interesting open problem is to convert our DBE scheme in composite-order bilinear groups to a DBE scheme in prime-order bilinear groups.

References

[1] Michel Abdalla, Eike Kiltz, and Gregory Neven. Generalized key delegation for hierarchical identity-based encryption. In Joachim Biskup and Javier Lopez, editors, *Computer Security - ESORICS 2007*, volume 4734 of *Lecture Notes in Computer Science*, pages 139–154. Springer, 2007.

[2] Nuttapong Attrapadung and Junichi Tomida. A modular approach to registered ABE for unbounded predicates. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology - CRYPTO 2024*, volume 14922 of *Lecture Notes in Computer Science*, pages 280–316. Springer, 2024.

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

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

[5] Dan Boneh and Mark Zhandry. Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology - CRYPTO 2014*, volume 8616 of *Lecture Notes in Computer Science*, pages 480–499. Springer, 2014.

[6] Jeffrey Champion, Yao-Ching Hsieh, and David J. Wu. Registered ABE and adaptively-secure broadcast encryption from succinct LWE. Cryptology ePrint Archive, Report 2025/44, 2025. `https://eprint.iacr.org/2025/044`.

[7] Jeffrey Champion and David J. Wu. Distributed broadcast encryption from lattices. In Elette Boyle and Mohammad Mahmoody, editors, *Theory of Cryptography - TCC 2024*, volume 15366 of *Lecture Notes in Computer Science*, pages 156–189. Springer, 2024.

[8] Melissa Chase and Sarah Meiklejohn. Déjà Q: Using dual systems to revisit q-type assumptions. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology - EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 622–639. Springer, 2014.

[9] Amos Fiat and Moni Naor. Broadcast encryption. In Douglas R. Stinson, editor, *Advances in Cryptology - CRYPTO '93*, volume 773 of *Lecture Notes in Computer Science*, pages 480–491. Springer, 1993.

[10] Rachit Garg, George Lu, Brent Waters, and David J. Wu. Realizing flexible broadcast encryption: How to broadcast to a public-key directory. In Weizhi Meng, Christian Damsgaard Jensen, Cas Cremers, and Engin Kirda, editors, *ACM Conference on Computer and Communications Security, CCS 2023*, pages 1093–1107. ACM, 2023.

[11] Rachit Garg, George Lu, Brent Waters, and David J. Wu. Reducing the CRS size in registered ABE systems. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology - CRYPTO 2024*, volume 14922 of *Lecture Notes in Computer Science*, pages 143–177. Springer, 2024.

[12] Sanjam Garg, Mohammad Hajiabadi, Mohammad Mahmoody, and Ahmadreza Rahimi. Registration-based encryption: Removing private-key generator from IBE. In Amos Beimel and Stefan Dziembowski, editors, *Theory of Cryptography - TCC 2018*, volume 11239 of *Lecture Notes in Computer Science*, pages 689–718. Springer, 2018.

[13] Sanjam Garg, Dimitris Kolonelos, Guru-Vamsi Policharla, and Mingyuan Wang. Threshold encryption with silent setup. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology - CRYPTO 2024*, volume 14926 of *Lecture Notes in Computer Science*, pages 352–386. Springer, 2024.

[14] Romain Gay, Lucas Kowalczyk, and Hoeteck Wee. Tight adaptively secure broadcast encryption with short ciphertexts and keys. In Dario Catalano and Roberto De Prisco, editors, *Security and Cryptography for Networks - SCN 2018*, volume 11035 of *Lecture Notes in Computer Science*, pages 123–139. Springer, 2018.

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

[16] Susan Hohenberger, George Lu, Brent Waters, and David J. Wu. Registered attribute-based encryption. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology - EUROCRYPT 2023*, volume 14006 of *Lecture Notes in Computer Science*, pages 511–542. Springer, 2023.

[17] Yao-Ching Hsieh, Brent Waters, and David J. Wu. A generic approach to adaptively-secure broadcast encryption in the plain model. In Serge Fehr and Pierre-Alain Fouque, editors, *Advances in Cryptology - EUROCRYPT 2025*, volume 15603 of *Lecture Notes in Computer Science*, pages 336–365. Springer, 2025.

[18] Dimitris Kolonelos, Giulio Malavolta, and Hoeteck Wee. Distributed broadcast encryption from bilinear groups. In Jian Guo and Ron Steinfeld, editors, *Advances in Cryptology - ASIACRYPT 2023*, volume 14442 of *Lecture Notes in Computer Science*, pages 407–441. Springer, 2023.

[19] Allison B. 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, 2010.

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

[21] 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 of *Lecture Notes in Computer Science*, pages 619–636. Springer, 2009.

[22] Hoeteck Wee. Déjà Q: Encore! un petit IBE. In Eyal Kushilevitz and Tal Malkin, editors, *Theory of Cryptography - TCC 2016-A*, volume 9563 of *Lecture Notes in Computer Science*, pages 237–258. Springer, 2016.

[23] Qianhong Wu, Bo Qin, Lei Zhang, and Josep Domingo-Ferrer. Ad hoc broadcast encryption. In Ehab Al-Shaer, Angelos D. Keromytis, and Vitaly Shmatikov, editors, *ACM Conference on Computer and Communications Security, CCS 2010*, pages 741–743. ACM, 2010.

[24] Ziqi Zhu, Kai Zhang, Junqing Gong, and Haifeng Qian. Registered ABE via predicate encodings. In Jian Guo and Ron Steinfeld, editors, *Advances in Cryptology - ASIACRYPT 2023*, volume 14442 of *Lecture Notes in Computer Science*, pages 66–97. Springer, 2023.