sensors

MDPI

Article

Backward Compatible Identity-Based Encryption

Jongkil Kim

Department of Cyber Security, Ewha Womans University, Seoul 03760, Republic of Korea; jongkil@ewha.ac.kr

Abstract: In this paper, we present a new identity-based encryption (IBE) system that is named Backward Compatible Identity-based Encryption (BC-IBE). Our BC-IBE is proposed to solve the problem caused by the out-of-synchronization between users' private keys and ciphertexts. Encryption systems such as revocable IBE or revocable Attribute-based Encryption (ABE) often require updating private keys to revoke users after a certain time period. However, in those schemes, an updated key can be used to decrypt the ciphertexts created only during the current time period. Once the key is updated and the previous keys are removed, the user, the owner of the updated key, will lose access to the past ciphertexts. In our paper, we propose BC-IBE that supports backward compatibility, to solve this problem. In our proposed system, user's private keys and ciphertexts can be updated periodically with time tags, and these processes can be used to revoke users who do not receive an updated key as the other revocable encryption does. However, in our proposed system, a private key newly issued to a user is backward compatible. This means that it decrypts not only the ciphertexts at the present time period but also all past ciphertexts. This implies that our proposed scheme guarantees the decryption of all encrypted data even if they are not synchronized. Compared to the existing revocable identity-based encryption system, our proposed BC-IBE has the advantage of simplifying key management and securely delegating ciphertext updates. Our proposed scheme only requires a single backward-compatible private key to decrypt all past ciphertexts created. Moreover, the ciphertext update process in our proposed scheme does not require any special privileges and does not require decryption. This means that this process can be securely delegated to a third-party server, such as a cloud server, and it prevents the potential leakage of secrets. For those reasons, BC-IBE is suitable for a system where users are more dynamic, such as the Internet-of-Things (IoT) network, or a system that regularly updates the data, like cloud data storage. In this paper, we provide the construction of BC-IBE and prove its formal security.

Keywords: identity-based encryption; public key encryption; revocation; IoT network security; cloud security

Citation: Kim, J. Backward Compatible Identity-Based Encryption. *Sensors* **2023**, *23*, 4181. https://doi.org/10.3390/s23094181

Academic Editor: Ilsun You

Received: 1 March 2023
Revised: 17 April 2023
Accepted: 20 April 2023
Published: 22 April 2023

1. Introduction

The Internet of Things (IoT) network is a network where a number of heterogeneous devices are connected to each other and exchange various types of data. As the data in an IoT network is often private, the security of the transmitted data in the IoT network is considered important. Public key encryption is the most widely used cryptographic system to control access to the IoT network in security protocols like Transport Layer Security (TLS), Datagram TLS, and Constraint Application Protocol (CoAP) because it does not require any pre-shared secret.

In such public key encryption systems, the authentication of devices is essential because the devices must check the identities of the corresponding party before encrypting a secret using its public key. Otherwise, the adversary can easily disguise as the other and hijack the data in the middle via an attack like Man-in-the-middle (MITM) attack [1,2]. Particularly, in a public key encryption system, a public key is transmitted through a non-secure channel. Therefore, a sender needs to check if the recipient's public key is matched with its identity so that it is truly from the same recipient who the sender intends to communicate with.

Public Key Infrastructure (PKI) is a widely used system for authentication and preventing this type of attack. A PKI system enables devices to authenticate if the recipient's identity and public key are matched before the sender sends any private data to the receiver. This conventional system implements the authentication process by issuing and maintaining certificates. Each certificate consists of identity, public keys and digital signatures so that it can be used to verify if the public key in the certificate belongs to the certificate owner via digital signatures. Unfortunately, maintaining certificates causes a significant burden to the system. It requires verifying multiple signatures for authentication and having to exchange large-sized certificates in addition to the cost of maintaining the certificate chain. Those are considered too large for the resource-constrained network such as IoT systems.

Identity-based encryption (IBE) [3–5] was introduced to manage access control based on the user's identity. In IBE, a sender encrypts a message with a public key that is associated with a receiver's identity. Furthermore, the receiver can decrypt ciphertexts with its private key that corresponds to the public key. The recipient's public key is not needed to be authenticated as it is already associated with its identity. Therefore, it reduces the burden of managing certificates that the IoT system wants to avoid due to its cost. Although an IBE system can bring a huge benefit to access control in an IoT system where a complex authentication scheme is needed due to its heterogeneity and scale, there are other properties that should be considered in an authentication. One of the properties that must be considered is dynamic access control.

In an IoT system, a user can join and be revoked from the system while the system is operating. One of the key features of IoT networks is sharing resources. A user can use multiple devices to share storage, as each individual device has relatively little storage. Cloud storage services such as Microsoft Onedrive and Google Drive are often used to share users' data across multiple devices, from desktop PCs to mobile phones or tablet PCs. However, those mobile devices can easily be lost and also out-of-synchronized by the carelessness of the user. This will put users' privacy at significant risk as those incidents will make a user's private key and data leak or become inaccessible from time to time.

In a traditional PKI system, a user (or its public key associated with the compromised private key) can be removed from its certificate chain by adding its certificate to a revocation list. However, in the IBE system, which does not use any certificates, the revocation is not straightforward. To support the revocation in IBE systems, Revocable IBE (RIBE) [6–10] was introduced to revoke a user when its private key is compromised.

RIBE uses a key-update mechanism. In RIBE, for a specific time slot, a user's long-term private key, for which the associated identity does not belong to the revocation list, can be used to compute the decryption key that can be used to decrypt the ciphertext encrypted in that time slot. Therefore, in the RIBE, one who does not have the decryption key in a specific time slot $\mathcal{T}$ is revoked. In RIBE, the decryption key cannot be used for the other time slots, particularly ciphertexts for the previous time slots. This may cause some compatibility problems in a practical system. For example, in the heterogeneous IoT network, some nodes are not synchronized properly and encrypt the data using the just previous time slot. The receiver cannot decrypt the data as it has already updated its key. Moreover, in cloud storage, the data is normally synchronized but not always. Some data, for example, stored in local storage, may not be synchronized properly. In this scenario, the data that are encrypted cannot be decrypted after the key update.

In this paper, we solve this problem, which we called the backward compatibility problem, by providing a new encryption system based on IBE that is named Backward Compatible IBE (BC-IBE). Trivial solutions for this problem are (1) keeping all previous decryption keys for backward compatibility or (2) decrypting and re-encrypting all ciphertexts created in the past time slots. However, the former requires more resources in secure memory, which is considered expensive. The latter requires a large overhead and the potential leakage of the secret as it causes decryption and re-encryption.

In our solution, BC-IBE allows a user to keep only a single key for the present time slot, but this key can be used to decrypt not only the ciphertexts created for the present

time slot where the current key is issued but also all past ciphertexts previously created. Moreover, at the same time, in our proposed BC-IBE, ciphertexts can be updated for a new time slot without decryption, so that it reduces the overhead caused by decryption and re-encryption.

1.1. Our Contribution

In this paper, we propose a new IBE encryption system, which is named backward compatible identity-based encryption (BC-IBE). Our proposed scheme provides the revocation of an expired private key via the update of ciphertexts and the backward compatibility of an updated private key. The details are as follows:

1. Backward compatibility: In our proposed scheme, a private key updated for time $\mathcal{T}$ can decrypt all the ciphertexts created at the time $\mathcal{T}$ and all the previous ciphertexts created before the time $\mathcal{T}$. One of the most trivial ways to achieve this is to keep all previous keys in secure storage. However, in that system, the user needs to maintain multiple keys to decrypt all past ciphertexts, which consumes large amounts of secure memory space. In our proposed scheme, the user only needs to maintain a single key at all times, but this key can be used to decrypt all past ciphertexts that were encrypted at previous time slots. At the same time, the same key can decrypt the ciphertext generated for the present time slot, too.
2. Revocation: In our proposed scheme, a private key is efficiently revoked by updating ciphertexts in the past time slots. That means that for all ciphertexts of the time $\mathcal{T}$, all previous keys issued before the time $\mathcal{T}$ cannot be used to decrypt the ciphertexts at time $\mathcal{T}$. Therefore, all past keys are revoked in the system.
3. Updating ciphertexts: In our BC-IBE, ciphertexts can be updated to revoke the past keys without decryption or re-encryption. Hence, the data owner does not need to download all ciphertexts and decrypt and re-encrypt them for the update. Moreover, this process does not involve any secret parameters. It can be easily delegated to a third party, such as a cloud service provider. Our scheme allows the updating of ciphertexts. This will be helpful for the overall scheme as (1) the scheme does not leak any information to the server while there is no secret involved in updating ciphertexts, (2) the server does not maintain the secret key for the ciphertext update and does not need any connection to be maintained to receive it.

We compare the above three properties of our BC-IBE scheme to the other encryption systems: identity-based encryption (IBE), revocable identity-based encryption (RIBE) and aggregate identity based encryption (AIBE) using Table 1. The plain IBE schemes [3–5,11,12] do not support revocation. As there are no states or separated time periods, backward compatibility is not applicable in the plain IBE. Due to the same reason, plain IBE does not support re-encrypting ciphertexts. RIBE [7,9,10,13,14] is a scheme for revoking invalid users only at a specific time slot. All ciphertexts in the past time slot cannot be decrypted using the decryption key at the present time slot, as each decryption key is a short-term key only for the current time slot. Therefore, it does not support the backward compatibility of a private key. The ciphertexts of the past time periods are accessible via decryption and re-encryption for the current time period.

AIBE [15] may be a suitable scheme for backward compatibility, as it can aggregate all past keys and the current key into a single one. However, it is not a scheme for revocation. Therefore, it does not provide revocation and re-encryption.

To summarize, our proposed scheme has the following contribution:

- We define the backward compatibility property of identity-based encryption and construct the first scheme that satisfies the backward compatibility property.
- Together with the backward compatibility property, our scheme supports the update of the ciphertexts. In our scheme, previous ciphertexts can be updated for the newly updated key without decryption, and they cannot be decrypted by all previously issued keys.

- As the encryption system we proposed is new, we newly develop the definition of BC-IBE, suggest its security model, and provide the security proof of the proposed scheme.

Table 1. Comparison of BC-IBE to the other encryption systems.

	Backward Compatibility	Revocation	Re-Encryption w/o Decryption
IBE [3–5,11,12]	N/A	No	No
RIBE [7,9,10,13,14]	No	Yes	No
AIBE [15]	Maybe	No	No
BC-IBE (Ours)	Yes	Yes	Yes

1.2. Paper Organization

The rest of this paper is organized as follows: in Section 1, we provide the introduction and contribution to explaining the motivation of our research. Sections 2 and 3 are for related work and preliminary explanations of the important literature for our work, respectively. In Section 4, the detailed method and the technical overview of our BC-IBE are explained. In Section 5, we present our construction and its formal security analysis. In Section 6. We discuss a potential threat and conclude our paper.

2. Related Works

The concept of identity-based encryption was proposed by Shamir [3]. It enables users to encrypt a message using their identities, such as e-mail, mobile numbers, and account numbers. Therefore, it reduces a lot of the burden needed to authenticate the receivers. The first practical IBE scheme was proposed by Boneh and Franklin [4] in a bilinear pairing group. Furthermore, it became an active research topic in public-key cryptography. Multiple IBE schemes [5,11,12] which improved security, were introduced, including adaptive security.

Revocable identity-based encryption (RIBE) is a system that improves an IBE scheme to support efficient revocation. As a user's private key can be compromised by an adversary, revocation is practically needed. The first practical RIBE scheme were introduced by Boldyreva et al. [13] using the complete subtree (CS) method, which achieves logarithm revocation complexity. Their scheme revokes a user's access by broadcasting an update key. The revoked users cannot compute a decryption key from the key update, so they are revoked. More schemes that achieve better security [14,16] or are based on different revocation methods [17] (e.g., the subset difference method) were introduced. Some RIBE systems are resistant to decryption key leakage. They are called decryption key exposure resistance (DKER) RIBE schemes [7,9,10]. Moreover, the generic constructions for RIBE from IBE was introduced by [6,8]. A few RIBE schemes also [18–20] use a third-party server to update the ciphertexts, similar to our scheme but without supporting backwards compatibility, and it also needs a key for the server to revoke users.

RIBE revokes users by updating decryption keys, but our scheme basically revokes users using the ciphertext and the status. This approach is more widely used in revocable attribute-based encryption (RABE) [21–23], in which each user has a private key that is associated with its attributes, and the decryption is allowed when its attributes satisfy a specific function (e.g., Boolean function). In those schemes, ciphertexts are re-encrypted or updated to revoke attributes to supporting dynamic credentials. The decryption key in RIBE can be used to decrypt the ciphertexts created within the same time slots. That is the main difference from our BC-IBE. In RIBE, the past ciphertexts can be updated only via decryption using the old keys, then it needs to be re-encrypted for the present time slot.

Improving the efficiency of those RIBE is still an ongoing problem. Lee et al. [24] and Yinxia et al. [25] presented RIBE schemes with a short key and a short ciphertext. More recently, Keita et al. [26] proposed a RIBE scheme that reduces the size of public keys.

However, those schemes focus on reducing the keys in the current time period and do not support backward compatibility. This means their scheme still needs a list of private keys for backward compatibility.

3. Preliminaries

3.1. Bilinear Pairing

Let's set $\mathcal{G}$ as a group generator that takes a security parameter λ as input and outputs a description of a bilinear group $\mathcal{G}$. For our purposes, we will have $\mathcal{G}$ output $(p, \mathbb{G}, \mathbb{G}_T, e)$ where p is a prime, $\mathbb{G}$ and $\mathbb{G}_T$ are cyclic groups of order p, and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is an efficiently computable non-degenerate bilinear map. We assume that the group operations in $\mathbb{G}$ and $\mathbb{G}_T$ as well as the bilinear map e are efficiently computable in polynomial time with respect to λ and that the group descriptions of $\mathbb{G}$ and $\mathbb{G}_T$ include generators of the respective cyclic groups.

3.2. Assumption

Assumption 1. *The q Decision Bilinear Diffie-Hellman Inversion (q-DBDHI) Assumption [15]. Let $\mathbb{G}$ and $\mathbb{G}_T$ be groups of order p with a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, and let g be a generator for $\mathbb{G}$. Set $\alpha \xleftarrow{R} \mathbb{Z}_p^*$ and $b \xleftarrow{R} \{0,1\}$. If $b = 0$, set $T \leftarrow e(g,g)^{1/\alpha}$; otherwise, set $T \xleftarrow{R} \mathbb{G}_T$. Output $\{g^{\alpha^i} : i \in [q]\}$ and T. The problem is to guess b.*

We define the advantage of the adversary of $\mathcal{A}$ to guess b correctly as follows:

$$Adv_{\mathcal{A},m}^{q-DBDHI}(\lambda) = \left| \Pr[b = b'] - \frac{1}{2} \right|.$$

3.3. Definitions

We write the formal definition of IBE using the syntax of [15].

Definition 1 (Identity-Based Encryption (IBE))**.** *The identity-based encryption for a set of identity spaces $\mathcal{I} = \{\{0,1\}^n\}_{n \in \mathbb{N}}$ and a message space $\mathcal{M}$ consists of the following PPT algorithm (Setup, KeyGen, Enc, Dec):*

- *Setup $(1^\lambda, 1^n) \to (pk, msk)$: takes as input the security parameter λ, the identity length n. It outputs the public parameters pk and the master secret key msk.*
- *KeyGen$(msk, id) \to (sk_{id})$: takes as input the master secret key msk and the identity $id \in \mathcal{I}$. It outputs a private key sk_{id}.*
- *Enc$(m, id, pk) \to (ct_{id})$: takes as input a message $m \in \mathcal{M}$, an identity $id \in \mathcal{I}$, and the public parameters pk. It outputs a ciphertext ct_{id}.*
- *Dec$(sk_{id}, ct_{id}) \to (m / \perp)$: takes as input a private key sk_{id}, a ciphertext ct_{id}. It outputs a message m or aborts.*

Correctness. An IBE scheme is correct if the following holds: for all $\lambda, n \in \mathbb{N}$, let $(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$, $\mathsf{sk}_{\mathsf{id}} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \mathsf{id})$ for $\mathsf{id} \in \{0,1\}^n$, and $\mathsf{ct}_{\mathsf{id}} \leftarrow \mathsf{Enc}(m, \mathsf{mpk}, \mathsf{id})$ for $m \in \mathcal{M}$. Furthermore, $m \leftarrow \mathsf{Dec}(\mathsf{sk}_{\mathsf{id}}, \mathsf{ct}_{\mathsf{id}})$.

The aggregating secret keys property, introduced in ref. [15], allows aggregating multiple private keys in an identity-based encryption scheme into a single compact key. They provide the definition of Aggregate Identity-Based Encryption(AIBE) by adding two extra algorithms to the definition of IBE as described in the following definition:

Definition 2 (Aggregate Identity-Based Encryption (AIBE))**.** *In addition to the algorithms (Setup, KeyGen, Enc, Dec) that forms an IBE scheme, the aggregating secret keys property requires the following two PPT algorithms (KeyAgg, AggDec) to support secret key aggregation in AIBE.*

1. *KeyAgg$(sk_1, \ldots sk_\ell) \to (\hat{sk})$ takes as input a sequence of secret keys $\{sk_i\}$ for $i \in [\ell]$ (for some $\ell > 1$). It outputs an aggregated $\hat{sk}$.*

2. *AggDec*($\hat{sk}$, ($id_1, \ldots id_\ell$), *ct*, *j*) $\rightarrow$ ($m/\perp$) *takes as input an aggregated private key* $\hat{sk}$*, a list of identities* $\{id_i \mid i \in [\ell]\}$*, a ciphertext* ct *and the index* $j \in [\ell]$ *that denotes the identity utilized to create* ct*. It outputs a message m or aborts.*

Correctness. An IBE scheme with aggregating secret keys is correct if the following holds: for all $\lambda, n \in \mathbb{N}$, let $(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$, identities $\mathsf{id}_i \in \{0,1\}^n$ for $i \in [\ell]$, every secret key $\mathsf{sk}_i \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \mathsf{id}_i)$ for $i \in [\ell]$, an aggregated secret key $\hat{\mathsf{sk}} \leftarrow \mathsf{KeyAgg}(\mathsf{sk}_1, \ldots, \mathsf{sk}_\ell)$ and a ciphertext $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{mpk}, \mathsf{id}_i, m)$. Furthermore, $m \leftarrow \mathsf{AggDec}(\hat{\mathsf{sk}}, (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathsf{ct}, j)$ for all $j \in [\ell]$, every message $m \leftarrow \mathcal{M}$.

Definition 3 (Security of IBE [15])**.** *The security of IBE scheme is defined as follows:*

- ***Setup*** *: The challenger runs* ***Setup***$(1^\lambda, 1^n)$ *to obtain a public key* pk*. It gives* $\mathcal{A}$ *the public key* pk*.*
- ***Phase I****: The adversary* $\mathcal{A}$ *requests* sk_{id_i} *for* $i \in \{1, \ldots, q_1\}$*.*
- ***Challenge****: If* ***Phase I*** *is over, the adversary* $\mathcal{A}$ *sends messages* m_0 *and* m_1 *with the challenge identity* id^* *to the challenger where* id^* *was not queried in* ***Phase I****. Furthermore, the challenger chooses a random binary* β *and runs* Enc *algorithm to calculate* $ct_{id^*} = Enc(m_\beta, id^*, pk)$ *and returns* (ct_{id^*}) *to* $\mathcal{A}$*.*
- ***Phase II****: The adversary* $\mathcal{A}$ *continues to requests private keys* sk_{id_i} *for* $i \in [q] \setminus [q_1]$*. For every pair* id_i *such that* $id_i \neq id^*$*, it returns* sk_{id_i} *to the adversary.*
- ***Guess****: Finally, the adversary* $\mathcal{A}$ *outputs a guess* $b' \in \{0,1\}$ *and wins the game if* $b = b'$*.*

We define the advantage of the adversary $\mathcal{A}$ to win in the game as following:

$$AdvIBE_{\mathcal{A},n}(\lambda) = \left| \Pr[b = b'] - \frac{1}{2} \right|.$$

Static security is a weaker notion of security. In static security, the adversary lets the challenger know all the identities to be queried and the challenge identity id^* before Setup. We use $AdvIBE^{Static}_{\mathcal{A},n}(\lambda)$ to denote the static security of IBE.

4. Our Method

The objective of our scheme is to build a more practical revocation system that supports ciphertext update and backward compatibility of the private key. Figure 1 depicts the backward compatibility that our scheme pursues. In the figure, tag_i implies a tag allocated for the ith time slot. The private key $\mathsf{sk}_{\mathsf{id},i}$ of an identity id for the time slot i can decrypt the ciphertext for the current time slot, $\mathsf{ct}_{\mathsf{id},i}$ and all past ciphertexts, $\mathsf{ct}_{\mathsf{id},j}$ for $j < i$. However, it cannot decrypt ciphertexts for future time slots.

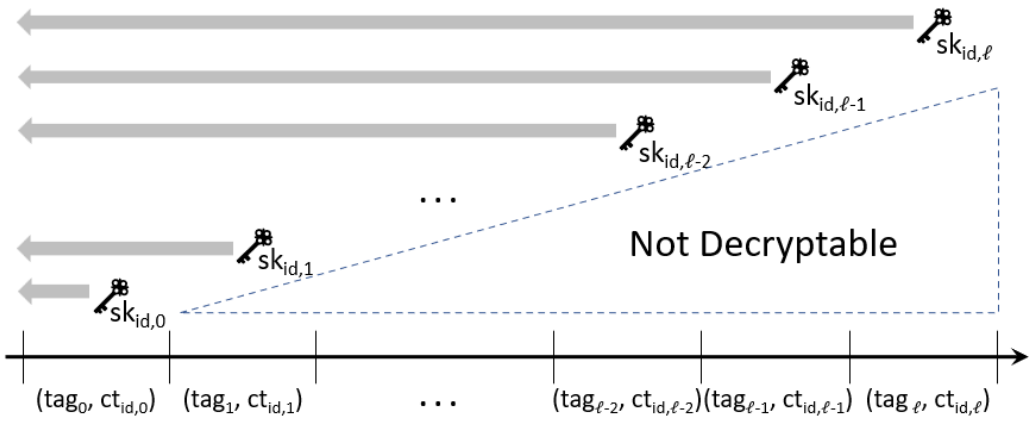

Figure 1. Backward Compatible Identity-Based Encryption (BC-IBE).

Our scheme is using a polynomial function to control access. Using a polynomial function is more popular for the variant of IBE, which is Identity-Based Broadcast Encryption (IBBE) [27–29]. IBBE is designed to share a single ciphertext for multiple users for broadcasting. As multiple identities are engaged in the encryption and decryption processes, it uses a polynomial function to handle this. In particular, in those schemes, the roots of a polynomial function are identities so it can be used to implement OR gates in the scheme.

In our scheme, a polynomial function is used differently. It has as its roots the identity of the recipient and tags that are uniquely allocated for time slots. So, one who can cancel out a polynomial function embedded in a ciphertext entirely using its private key that is associated with the identity and the tags can decrypt the ciphertext. It means that the polynomial function works in our scheme as AND gates.

For example, let a, t_1 and t_2 be the identity and the tags used for Alice, respectively. The ciphertext is constructed by using a polynomial function $P(a, t_1, t_2) = (x - a)(x - t_1)(x - t_2)$ (e.g., $g^{r \cdot P(a,t_1,t_2)}$ where g is a group generator and r is a randomization parameter.) and Alice has a private key computed based on $1/P(a, t_1, t_2)$ (e.g., $g^{1/P(a,t_1,t_2)}$) at the second time slot. Therefore, Alice can decrypt the ciphertext using a pairing computation (e.g., $e(g^{r \cdot P(a,t_1,t_2)}, g^{1/P(a,t_1,t_2)}) = e(g, g)^r$) using her key. Therefore, decryption is possible only for one who has a private key based on the inverse of the polynomial function given in the ciphertext.

For the revocation, we observed that, although a polynomial function is set in a ciphertext, it can be updated to become more restrictive. Using the previous example, if the following is given

$$(C = m \cdot e(g, g)^r, C_2 = g^{r \cdot P(a,t_1,t_2)x^2}, C_1 = g^{r \cdot P(a,t_1,t_2)x}, C_0 = g^{r \cdot P(a,t_1,t_2)}).$$

To update the ciphertext with t_3, the tag for the third time slot is given, and one can update the ciphertext as

$$C' = m \cdot e(g, g)^r, C_2' = R,\ C_1' = C_2 \cdot C_1^{-t_3} = g^{r \cdot P(a,t_1,t_2)(x-t_3)x} = g^{r \cdot P(a,t_1,t_2,t_3)x},$$

$$C_0' = C_1 \cdot C_0^{-t_3} = g^{rP(a,t_1,t_2)(x-t_3)} = g^{r \cdot P(a,t_1,t_2,t_3)}$$

where R is a random value and $P(a, t_1, t_2, t_3) = (x - a)(x - t_1)(x - t_2)(x - t_3)$. This can be completed without any other secret parameter and does not need decryption. Moreover, once the new ciphertext is updated successfully, it cannot go back.

For the updated ciphertext, all previous keys such as $g^{1/P(a,t_1,t_2)}$ are no longer valid as they cannot completely divide the polynomial function given in the ciphertext due to $(x - t_3)$. Therefore, they are revoked. However, the newly generated key at the time t_3 can be used for all previous ciphertexts by executing the update process locally. Therefore, this method can be used to guarantee backward compatibility.

To prove the security of our scheme, we utilize aggregate identity-based encryption (AIBE) . AIBE was introduced by Goyal and Vaikuntanathan [15]. Their scheme is used for a system that maintains multiple private keys. It uses the key accumulate algorithm, called DPP, from Delerablée, Paillier and Pointcheval [30,31] to aggregate multiple keys such as $\{g^{1/(x-a_i)}, a_i\}_{i \in [\ell]}$ into a single key $g^{1/(x-a_1)(x-a_2)\cdots(x-a_\ell)}$. Our scheme does not need key aggregation as a private key can be updated directly by a key-update key. However, it is still useful to prove the security of BC-IBE, as we can query the private keys and use them to form a key-update key without knowing the master secret.

Before we present our main construction, we provide the formal definitions of our BC-IBE and its security models in the following subsections.

4.1. Definition of BC-IBE

Using the notion of identity-based encryption, we provide the definition of Backward Compatible Identity-Based Encryption (BC-IBE). Our BC-IBE consists of seven algorithms, Setup, UpdateKeyGen, KeyGen, KeyUpdate, Enc, EncUpdate, and Dec, as defined below:

- $\mathsf{Setup}(1^\lambda, 1^{n_1}, 1^{n_2}, 1^{n_3}) \rightarrow (\mathsf{pk}, \mathsf{msk})$: takes as input the security parameter λ, the identity length n_1, the tag length n_2 and the maximum number of updates n_3. It outputs a public key pk and a master secret key msk.
- $\mathsf{UpdateKeyGen}((\mathsf{tag}_1, \ldots, \mathsf{tag}_\ell), \mathsf{id}, \mathsf{msk}) \rightarrow (\mathsf{usk}_{\mathsf{id},\ell})$: takes as input a sequence of tags $\mathsf{tag}_i \in \{0,1\}^{n_2}$ for $i \in [\ell]$, an identity $\mathsf{id} \in \{0,1\}^{n_1}$ and the secret key msk. It outputs a key-update key $\mathsf{usk}_{\mathsf{id},\ell}$.
- $\mathsf{KeyGen}(\mathsf{id}, \mathsf{msk}) \rightarrow \mathsf{sk}_{\mathsf{id},0}$: takes as input an identity id and the master secret key msk. It outputs a private key $\mathsf{sk}_{\mathsf{id},0}$.
- $\mathsf{KeyUpdate}(\mathsf{sk}_{\mathsf{id},j-1}, \mathsf{usk}_{\mathsf{id},j}) \rightarrow \mathsf{sk}_{\mathsf{id},j}$: takes as input the private key $\mathsf{sk}_{\mathsf{id},j-1}$ and the update key $\mathsf{usk}_{\mathsf{id},j}$ for $j \leq n_3$. It outputs a private key $\mathsf{sk}_{\mathsf{id},j}$.
- $\mathsf{Enc}(m, \mathsf{id}, \mathsf{pk}) \rightarrow \mathsf{ct}_0$: takes as input a message $m \in \mathcal{M}$, an identity $\mathsf{id} \in \{0,1\}^{n_2}$, and a public key pk. It outputs a ciphertext ct_0.
- $\mathsf{EncUpdate}(\mathsf{pk}, \mathsf{ct}_{\mathsf{id},j-1}, \mathsf{id}, (\mathsf{tag}_1, \ldots, \mathsf{tag}_j)) \rightarrow \mathsf{ct}_{\mathsf{id},j}$: takes as input the ciphertext $\mathsf{ct}_{\mathsf{id},j-1}$, an identity id, a sequence of tags $(\mathsf{tag}_1, \ldots, \mathsf{tag}_j)$. The algorithm outputs the updated ciphertext $\mathsf{ct}_{\mathsf{id},j}$.
- $\mathsf{Dec}(\mathsf{sk}_{\mathsf{id},i}, \mathsf{ct}_{\mathsf{id},j}) \rightarrow m/\perp$: takes as input a private key $\mathsf{sk}_{\mathsf{id},i}$, a ciphertext $\mathsf{ct}_{\mathsf{id},j}$. It outputs the message m or aborts.

Correctness. For the correctness of a BC-IBE scheme, the following property must be satisfied: for all $\lambda, n_1, n_2, n_3 \in \mathbb{N}$, let $(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^{n_1}, 1^{n_2}, 1^{n_3})$, $\mathsf{sk}_{\mathsf{id},0} \leftarrow \mathsf{KeyGen}(\mathsf{id}, \mathsf{msk})$ for every identity $\mathsf{id} \in \{0,1\}^{n_1}$ and $\mathsf{ct}_{\mathsf{id},0} \leftarrow \mathsf{Enc}(m, \mathsf{id}, \mathsf{pk})$. $m \leftarrow \mathsf{Dec}(\mathsf{sk}_{\mathsf{id},0}, \mathsf{ct}_{\mathsf{id},0})$ for every message m.

Correctness for updated ciphertexts and keys. For correctness for updated ciphertexts and keys of a BC-IBE scheme, the following property must be satisfied: for all $\lambda, n_1, n_2, n_3 \in \mathbb{N}$, let $(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^{n_1}, 1^{n_2}, 1^{n_3})$, $\mathsf{sk}_{\mathsf{id},0} \leftarrow \mathsf{KeyGen}(\mathsf{id}, \mathsf{msk})$ for every identity $\mathsf{id} \in \{0,1\}^{n_1}$. For the update keys, let $\mathsf{usk}_{\mathsf{id},i} \leftarrow \mathsf{UpdateKeyGen}((\mathsf{tag}_1, \ldots, \mathsf{tag}_i), \mathsf{id}, \mathsf{msk})$ with tags $\mathsf{tag}_1, \ldots \mathsf{tag}_i \in \{0,1\}^{n_2}$ for any $i \leq n_3$ and $\mathsf{sk}_{\mathsf{id},i} \leftarrow \mathsf{KeyUpdate}(\mathsf{sk}_{\mathsf{id},i-1}, \mathsf{usk}_{\mathsf{id},i})$ that is repeatedly computed. Furthermore, let $\mathsf{ct}_{\mathsf{id},0} \leftarrow \mathsf{Enc}(m, \mathsf{id}, \mathsf{pk})$ and $\mathsf{ct}_{\mathsf{id},j} \leftarrow \mathsf{EncUpdate}$ $(\mathsf{pk}, \mathsf{ct}_{\mathsf{id},j-1}, \mathsf{id}, (\mathsf{tag}_1, ..., \mathsf{tag}_j))$ for $j \leq n_3$. Furthemore, $m \leftarrow \mathsf{Dec}(\mathsf{sk}_{\mathsf{id},i}, \mathsf{ct}_{\mathsf{id},j})$ for all $j \leq i$ for every message m.

4.2. Security of BC-IBBE

We define the security of our BC-IBE scheme as follows:

- **Setup**: The challenger runs $\mathsf{Setup}(1^\lambda, 1^{n_1}, 1^{n_2}, 1^{n_3})$ to obtain a public key pk. It gives $\mathcal{A}$ the public key pk.
- **Phase I**: The adversary $\mathcal{A}$ issues the following queries to the challenger:
  1. For $i \in [q_1]$, it requests $\mathsf{sk}_{\mathsf{id}_i,0}$.
  2. For $i \in [q_1]$, the challenger sets $\mathsf{tag}_1, \ldots, \mathsf{tag}_k \in \{0,1\}^{n_2}$ for $k \in [n_3]$ and it requests $\mathsf{usk}_{\mathsf{id}_i,k}$ for $(\mathsf{id}_i, \{\mathsf{tag}_j\}_{j\in[k]})$.

  For each query, the challenger returns the resulting key to the adversary.
- **Challenge**: When **Phase I** is over, the adversary $\mathcal{A}$ sends messages m_0 and m_1 with a challenge of identity id^* and tags, $(\mathsf{tag}_1^*, \ldots, \mathsf{tag}_\ell^*)$ in that the pair $(\mathsf{id}^*, \mathsf{tag}_\ell^*)$ has never been queried together in **Phase I** to the challenger. The challenger chooses a random binary β and runs the Enc algorithm to compute $\mathsf{ct}_{\mathsf{id}^*,0} = \mathsf{Enc}(m_\beta, \mathsf{id}^*, \mathsf{pk})$, and then it updates the ciphertext $\mathsf{ct}_{\mathsf{id}^*,0}$ to the ciphertext $\mathsf{ct}_{\mathsf{id}^*,j}$ using EncUpdate by executing consecutively $\mathsf{EncUpdate}(\mathsf{pk}, \mathsf{ct}_{\mathsf{id}^*,i-1}, \mathsf{id}^*, (\mathsf{tag}_1^*, \ldots, \mathsf{tag}_i^*))$ for $i \in [\ell]$. The challenger returns $(\mathsf{ct}_{\mathsf{id}^*,\ell})$ to $\mathcal{A}$.
- **Phase II**: The adversary $\mathcal{A}$ continues to issue the following queries:
  1. For $i \in [q] \setminus [q_1]$, it requests $\mathsf{sk}_{\mathsf{id}_i,0}$.

2. For $i \in [q] \setminus [q_1]$, the challenger sets $\mathsf{tag}_1, \ldots, \mathsf{tag}_k \in \{0,1\}^{n_2}$ such that $k \in [n_3]$ and it requests $\mathsf{usk}_{\mathsf{id}_i,k}$ for $(\mathsf{id}_i, \{\mathsf{tag}_j\}_{j\in[k]})$ with the restriction that the challenge pair $(\mathsf{id}^*, \mathsf{tag}^*_\ell)$ cannot be queried.

For each query, the challenger forwards the resulting key to the adversary.

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

We define the advantage of the adversary $\mathcal{A}$ to win in the game as follows:

$$\mathit{AdvBC\text{-}IBE}_{\mathcal{A},n}(\lambda) = \left| \Pr[b = b'] - \frac{1}{2} \right|.$$

Static security is the weaker security notion of BC-IBE by adding the step that the adversary lets the challenger know all identities and tags to be queried before Setup (i.e., before requesting any parameters from the challenger). We use $\mathit{AdvBC\text{-}IBE}^{Static}_{\mathcal{A},n}(\lambda)$ to denote the advantage of the adversary in the static security model.

5. Our Result

5.1. Our Construction

Let $\mathcal{G}(\lambda)$ be an algorithm that outputs bilinear group parameters $\langle \mathbb{G}, \mathbb{G}_T, e \rangle$, where $\mathbb{G}$ and $\mathbb{G}_T$ are of order p, and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. Let g and $e(g,g)$ be generators of $\mathbb{G}$ and $\mathbb{G}_T$, respectively. With a collision-resistant hash algorithm $H : \{0,1\}^{n_1} \times \{0,1\}^{n_2} \to \mathbb{Z}_p$ where n_1 and n_2 are the lengths of identities and tags, respectively. Furthermore, n_3 is the maximum number of updates. The construction of our BC-IBE scheme is as follows:

- $\mathsf{Setup}(1^\lambda, 1^{n_1}, 1^{n_2}, 1^{n_3}) \to (\mathsf{pk}, \mathsf{msk})$ takes the security parameter λ as input and the sizes of identities (n_1) and tags (n_2) together with the maximum number of the key updates (n_3). Furthermore, the algorithm generates the bilinear group $\langle p, \mathbb{G}, \mathbb{G}_T, g, e \rangle \leftarrow \mathcal{G}(1^\lambda)$. It randomly chooses parameters α and β in $\mathbb{Z}_p$ and $\mathsf{hk} \leftarrow \mathsf{HGen}(1^\lambda)$ where hk is a parameter for the identity hash algorithm. It sets a public key $\mathsf{pk} := (\mathsf{hk}, g, \{g^{\beta^i}\}_{i\in[n_3]}, g^\alpha)$ and a private key $\mathsf{msk} := (\mathsf{hk}, \alpha, \beta)$.
- $\mathsf{UpdateKeyGen}((\mathsf{tag}_1, \ldots, \mathsf{tag}_\ell), \mathsf{id}, \mathsf{msk}) \to (\mathsf{usk}_{\mathsf{id},\ell})$: takes the set of tags $(\mathsf{tag}_1, \ldots \mathsf{tag}_\ell)$ as inputs, an identity id and the master secret key msk. It sets $h_{\mathsf{id},i} = H(\mathsf{hk}, \mathsf{id}, \mathsf{tag}_i)$ for $i \in [\ell]$. It sets $\mathsf{usk}_{\mathsf{id},\ell} = g^{\alpha/((\beta+h_{\mathsf{id},0})\cdots(\beta+h_{\mathsf{id},\ell}))-\alpha/((\beta+h_{\mathsf{id},0})\cdots(\beta+h_{\mathsf{id},\ell-1}))}$ where $h_{\mathsf{id},0} = H(\mathsf{hk}, \mathsf{id}, 0)$ for $1 \le \ell$. It outputs $\mathsf{usk}_{\mathsf{id},\ell}$.
- $\mathsf{KeyGen}(\mathsf{id}, \mathsf{msk}) \to \mathsf{sk}_{\mathsf{id},0}$: The key generation algorithm takes the identity id and the master secret key msk. It then outputs $\mathsf{sk}_{\mathsf{id},0} = g^{\alpha/(\beta+h_{\mathsf{id},0})}$ where $h_{\mathsf{id},0} = H(\mathsf{hk}, \mathsf{id}, 0)$.
- $\mathsf{KeyUpdate}(\mathsf{sk}_{\mathsf{id},\ell-1}, \mathsf{usk}_{\mathsf{id},\ell}) \to \mathsf{sk}_{\mathsf{id},\ell}$: The key generation algorithm takes the latest updated key $\mathsf{usk}_{\mathsf{id},\ell}$ and the secret key $\mathsf{sk}_{\mathsf{id},\ell-1}$ for $1 \le \ell$. It updates the secret key

$$\mathsf{sk}_{\mathsf{id},\ell} = \mathsf{sk}_{\mathsf{id},\ell-1} \cdot \mathsf{usk}_{\mathsf{id},\ell}.$$

It should be noted that the above equation results in $\mathsf{sk}_{\mathsf{id},\ell} = g^{\alpha/\prod_{i=0}^{\ell}(\beta+h_{\mathsf{id},\ell})}$. It returns $\mathsf{sk}_{\mathsf{id},\ell}$.

- $\mathsf{Enc}(m, \mathsf{pk}, \mathsf{id}) \to \mathsf{ct}_{\mathsf{id},0}$: The encryption algorithm takes a message m, a public key pk and an identity id. It randomly selects a random value $r \in \mathbb{Z}_p$ and computes $h_{\mathsf{id},0} = H(\mathsf{hk}, \mathsf{id}, 0)$. It, then, sets the following as the ciphertext ct:

$$C_i := g^{r(\beta+h_{\mathsf{id},0})\beta^i}, C_T := m \cdot e(g,g)^{\alpha\cdot r}$$

It outputs $\mathsf{ct}_{\mathsf{id},0} := (\{C_i\}_{i=0}^{n_3}, C_T)$.

- $\mathsf{EncUpdate}(\mathsf{pk}, \mathsf{ct}_{\mathsf{id},\ell-1}, \ell, \mathsf{id}, (\mathsf{tag}_1, \ldots, \mathsf{tag}_\ell)) \to (\mathsf{ct}_{\mathsf{id},\ell})$: The re-encryption algorithm takes a public key pk and a ciphertext $\mathsf{ct}_{\mathsf{id},\ell-1}$ and the identity id and a sequence of the tags $(\mathsf{tag}_1, \ldots, \mathsf{tag}_\ell)$. It randomly selects a random value $r' \in \mathbb{Z}_p$ and computes $(h_{\mathsf{id},0}, h_{\mathsf{id},1}, \ldots, h_{\mathsf{id},\ell}) = (H(\mathsf{hk}, \mathsf{id}, 0), H(\mathsf{hk}, \mathsf{id}, \mathsf{tag}_1), \ldots, H(\mathsf{hk}, \mathsf{id}, \mathsf{tag}_\ell))$. It com-

putes $(a_0, a_1, \ldots, a_{\ell+1})$ where a_j is the coefficient of x^j in the polynomial function $(x + h_{\mathsf{id},0})(x + h_{\mathsf{id},1})(x + h_{\mathsf{id},2}) \cdots (x + h_{\mathsf{id},\ell})$. It computes $C_i'' = \{g^{\sum_{j=0}^{\ell+1} a_j \cdot \beta^{i+j}}\}_{i=0}^{n_3-\ell}$. It sets $C_T' := C_T \cdot e(g,g)^{\alpha r'}$. For all $i \in [n_3 - \ell]$,

$$C_i' := C_{i+1} \cdot {C_i}^{h_{\mathsf{id},i}} \cdot C_i''$$

For all i such that $n_3 - \ell < i \leq n_3$, it randomly selects $R_i \in \mathbb{G}$ also sets $C_i' := C_i \cdot R_i$. It outputs the updated ciphertext:

$$\mathsf{ct}_{\mathsf{id},\ell} =: (\{C_i'\}_{i=0}^{n_3}, C_T')$$

- $\mathsf{Dec}(\mathsf{sk}_{\mathsf{id},\ell}, \mathsf{ct}_{\mathsf{id},j}, (\mathsf{tag}_1, \ldots, \mathsf{tag}_\ell), j, \mathsf{hk}) \to (m)$: The decryption algorithm takes the secret key $\mathsf{sk}_{\mathsf{id},\ell}$ and the ciphertext $\mathsf{ct}_{\mathsf{id},j}$ of id and the tags $\{\mathsf{tag}_1, \ldots \mathsf{tag}_\ell\}$. Furthermore, we set coefficient a_i as follows:
  - If $j = \ell$, $a_0 = 1$.
  - If $j < \ell$, for all $i \in \{0, \ldots, \ell - j\}$, a_i is the coefficients of x^i of the polynomial $(x + h_{\mathsf{id},j+1}) \cdots (x + h_{\mathsf{id},\ell})$ where $h_{\mathsf{id},j} = H(\mathsf{hk}, \mathsf{id}, \mathsf{tag}_j)$.

  It, then, parses $\mathsf{ct}_{\mathsf{id},j}$ to $C_0 \ldots C_{n_3}$ and C_T and computes

$$m = C_T \cdot e(\prod_{i=0}^{\ell-j} {C_i}^{a_i}, \mathsf{sk}_{\mathsf{id},\ell})^{-1}.$$

Correctness. By the definition, $C_0 = g^{r(\beta + h_{\mathsf{id},0})}$, $sk_{\mathsf{id},0} = g^{\alpha/(\beta + h_{\mathsf{id},0})}$ and $C_T = m \cdot e(g,g)^{\alpha r}$. Therefore,

$$\begin{aligned} C_T \cdot e(C_0, \mathsf{sk}_{\mathsf{id},0})^{-1} &= m \cdot e(g,g)^{\alpha r} \cdot e(g^{r \cdot (\beta + h_{\mathsf{id},0})}, g^{\alpha/(\beta + h_{\mathsf{id},0})})^{-1} \\ &= m \cdot e(g,g)^{\alpha \cdot r} \cdot e(g,g)^{-\alpha \cdot r} \\ &= m \end{aligned}$$

Correctness for updated ciphertexts and keys. For $j \leq \ell$, by the definition, $C_j = g^{\hat{r} \prod_{i=0}^{j} (\beta + h_{\mathsf{id},i})}$, $sk_{\mathsf{id},\ell} = g^{\alpha / \prod_{i=1}^{\ell} (\beta + h_{\mathsf{id},i})}$ and $C_T' = m \cdot e(g,g)^{\alpha \cdot \hat{r}}$. First, we compute a_i that is the coefficient of x^i of the polynomial $(x + h_{\mathsf{id},j+1}) \cdots (x + h_{\mathsf{id},\ell})$ for all $i \in \{0, \ldots, \ell - j\}$. Therefore,

$$\begin{aligned} C_T' \cdot e(\prod_{i=0}^{\ell-j} {C_i}^{a_i}, \mathsf{sk}_{\mathsf{id},\ell})^{-1} &= m \cdot e(g,g)^{\alpha \hat{r}} \cdot e(\prod_{i=0}^{\ell-j} C_i^{a_i}, \mathsf{sk}_{\mathsf{id},\ell})^{-1} \\ &= m \cdot e(g,g)^{\alpha \cdot \hat{r}} \cdot e(g^{\hat{r} \cdot (\prod_{i=0}^{j} (\beta + h_{\mathsf{id},i}))(\sum_{i=0}^{\ell-j} a_i \cdot \beta^i)}, g^{\alpha / \prod_{i=0}^{\ell} (\beta + h_{\mathsf{id},i})}) \\ &= m \cdot e(g,g)^{\alpha \cdot \hat{r}} \cdot e(g^{\hat{r} \cdot (\prod_{i=0}^{\ell} (\beta + h_{\mathsf{id},i}))}, g^{\alpha / \prod_{i=0}^{\ell} (\beta + h_{\mathsf{id},i})}) \\ &= m \cdot e(g,g)^{\alpha \cdot \hat{r}} \cdot e(g,g)^{-\alpha \cdot \hat{r}} \\ &= m \end{aligned}$$

Theorem 1. *Our BC-IBE is static secure under (q-DBDHI) assumption.*

Proof. We will show the security of our BC-IBE using Lemma 1 in our security analysis. It will be proven to be secure by showing the oracles simulating the security of the AIBE from Goyal and Vaikuntanathan are invariant. Therefore, it will have the same security that AIBE has. □

5.2. Security Analysis

We utilize Goyal and Vaikuntanathan's AIBE (see Appendix A) to prove our scheme. Their AIBE scheme is static secure and its static security is proven under q-DBDHI assump-

tion in a random oracle model. In our scheme, we also show that our scheme is secure via the security of the AIBE scheme.

To use the AIBE scheme for the security proof of BC-IBE, we define two indistinguishable oracles $\mathcal{O}^0_{AIBE}$ and $\mathcal{O}^1_{AIBE}$ that simulate the static security of AIBE. First, we define $\mathcal{O}^0_{AIBE}$ as follows:

- $\mathsf{Setup}(1^\lambda, 1^n, \mathsf{id}^*) \rightarrow (\mathsf{mpk})$: takes as input the security parameter λ, the identity length n. It returns the public parameters $\mathsf{mpk} = (\mathsf{hk}, \{g^{\beta^i}\}_{i\in[n]})$ where $\mathsf{hk} \leftarrow \mathsf{HGen}(1^\lambda)$ is a parameter for the hash algorithm for identities H (i.e., $H(\mathsf{hk}, \mathsf{id}) = h_{\mathsf{id}}) \in \mathbb{Z}_p$).
- $\mathsf{Query}(\mathsf{id} \in \{0,1\}^n \setminus \{\mathsf{id}^*\}) \rightarrow (\mathsf{sk}_{\mathsf{id}})$: When a secret key $\mathsf{sk}_{\mathsf{id}}$ for the identity $\mathsf{id} \in \{0,1\}^n$ is requested, it returns a private key $\mathsf{sk}_{\mathsf{id}} = g^{(\beta+h_{\mathsf{id}})^{-1}}$ where $h_{\mathsf{id}} = H(\mathsf{hk}, \mathsf{id})$.
- $\mathsf{Challenge}(m_0, m_1, \mathsf{id}^*, \mathsf{mpk}) \rightarrow (\mathsf{ct}_{\mathsf{id}^*})$ takes as input messages $m_0, m_1 \in \mathcal{M}$, an identity $\mathsf{id}^* \in \{0,1\}$, and the public parameters mpk. It randomly selects $b \in \{0,1\}$ and outputs the challenge ciphertext

$$\mathsf{ct}_{\mathsf{id}^*} = \left(\{g^{r(\beta+h_{\mathsf{id}^*})\beta^i}\}_{i=0}^n, m_b \cdot e(g,g)^r\right)$$

where r is a randomly selected value in $\mathbb{Z}_p$.

The oracle $\mathcal{O}^1_{AIBE}$ is defined identically except that m_b in $\mathsf{ct}_{\mathsf{id}^*}$ is replaced by a random message in $\mathcal{M}$. It should be noted that the oracles $\mathcal{O}^0_{AIBE}$ and $\mathcal{O}^1_{AIBE}$ are indistinguishable under $q-DBDHI$ assumption in the random oracle model by the static security of Aggregate IBE in [15].

Lemma 1. *Suppose there is a PPT algorithm $\mathcal{A}$ that breaks the static security of BC-IBE with non-negligible probability ϵ. Furthermore, we can build an algorithm $\mathcal{B}$ that distinguishes between $\mathcal{O}^0_{AIBE}$ and $\mathcal{O}^1_{AIBE}$ using $\mathcal{A}$ with ϵ.*

Proof. We are going to prove the static security of our BC-IBE scheme using the indistinguishability between $\mathcal{O}^0_{AIBE}$ and $\mathcal{O}^1_{AIBE}$ as follows:

Before Setup, for the initialization, the challenger sets the identities to be queried and the maximum number of updates n_3. It also chooses the identity to be challenged, id^* and the number of updates to be challenged for id^*, which is tag^*_ℓ. The algorithm $\mathcal{B}$ sets a new identity space that is defined as

$$\{\{\mathsf{id}_i||\mathsf{tag}_{i,j}\}_{i\in[q],j\in[n_3]},\ \{\mathsf{id}^*||\mathsf{tag}_j\}_{j\in[n_3]}\}$$

and the challenge identity $\mathsf{id}^*||\mathsf{tag}^*_\ell$. $\mathcal{B}$ sends them to the oracle (either $\mathcal{O}^0_{AIBE}$ or $\mathcal{O}^1_{AIBE}$) that it works with. The oracle uses this for the initialization by creating $n_1 + n_2$ sized identity space (i.e., $\{0,1\}^{n_1+n_2}$ where n_1 and n_2 are the sizes of the identity and tag spaces, respectively. Through this process, $\mathcal{B}$ will break the static security of AIBE by distinguishing between $\mathcal{O}^0_{AIBE}$ and $\mathcal{O}^1_{AIBE}$ using $\mathcal{A}$.

Setup: To create pk for $\mathcal{B}$. $\mathcal{B}$ requests a public key to the oracle, it works with. The oracle sends $(\mathsf{hk}, \{g^{\beta^i}\}_{i\in[n]})$ back to $\mathcal{B}$. Furthermore, $\mathcal{B}$ randomly selects α and sends $(H, \{g^{\beta^i}\}_{i\in[n]}, g^\alpha)$ as pk where H is a random oracle hashing identities with $\mathsf{hk} \leftarrow \mathsf{HGen}(1^\lambda)$ by concatenating an identity and a tag and taking it as input together with hk. (i.e., $H(\mathsf{hk}, \mathsf{id}||\ \mathsf{tag})$).

Phase I/II: In this stage, $\mathcal{A}$ can query three types of queries and $\mathcal{B}$ responds it as follows:

1. When $\mathcal{A}$ requests $\mathsf{sk}_{\mathsf{id}_i,0}$ for $i \in [q]$, $\mathcal{B}$ requests the private key for $\mathsf{id}_i||0$ to the oracle. $\mathcal{B}$ receives $\hat{\mathsf{sk}}_{\mathsf{id}_i||0}$ from the oracle. Furthermore, it sets $\mathsf{sk}_{\mathsf{id}_i,0} = (\hat{\mathsf{sk}}_{\mathsf{id}_i||0})^\alpha$ and returns it to $\mathcal{A}$.
2. When $\mathcal{A}$ requests $\mathsf{usk}_{\mathsf{id}_i,j}$ for $\mathsf{id}_i||\mathsf{tag}_{i,j}$ for $i \in [q], j \in [n_3]$, $\mathcal{B}$ requests the a private key for $\mathsf{id}_i||\mathsf{tag}_{i,j}$ to the oracle. $\mathcal{B}$ receives $\hat{\mathsf{sk}}_{\mathsf{id}_i||\mathsf{tag}_{i,j}}$ from the oracle it works

with. Furthermore, it computes $d_1 = \mathsf{DPP}(\hat{\mathsf{sk}}_{\mathsf{id}_i||0}, \hat{\mathsf{sk}}_{\mathsf{id}_i||\mathsf{tag}_{i,1}}, \ldots, \hat{\mathsf{sk}}_{\mathsf{id}_i||\mathsf{tag}_{i,j-1}})$ and $d_2 = \mathsf{DPP}(\hat{\mathsf{sk}}_{\mathsf{id}_i||0}, \hat{\mathsf{sk}}_{\mathsf{id}_i||\mathsf{tag}_{i,1}}, \ldots, \hat{\mathsf{sk}}_{\mathsf{id}_i||\mathsf{tag}_{i,j}})$. It sets $\mathsf{usk}_{\mathsf{id}_i,j} = (d_2)^{\alpha} - (d_1)^{\alpha}$ and returns it to $\mathcal{A}$ where the function DPP [30,31] is an aggregate algorithm that can aggregate multiple keys such as $\{g^{1/(x+a_i)}, a_i\}_{i\in[\ell]}$ into a single key $g^{1/(x+a_1)(x+a_2)\cdots(x+a_\ell)}$.

3. $\mathcal{A}$ also can request $\mathsf{sk}_{\mathsf{id}^*||0}$ and $\mathsf{usk}_{\mathsf{id}^*||\mathsf{tag}_i^*}$ for $i \in [\ell - 1]$ and also $\mathcal{B}$ can respond in the same way it responds above. They cannot query $\mathsf{usk}_{\mathsf{id}^*||\mathsf{tag}_\ell^*}$.

Challege: When $\mathcal{A}$ requests the challenge ciphertext for id^*, $(\mathsf{tag}_1^*, \ldots, \mathsf{tag}_\ell^*)$, $\mathcal{B}$ requests the challenge ciphertext of $\mathsf{id}^*||\mathsf{tag}_\ell^*$ to the oracle that it works with. The oracle will send the challenge ciphertext that is either a ciphertext for m_b or a random message following:

$$\mathsf{ct} = \left(\{g^{r(\beta+h_{\mathsf{id}^*||\mathsf{tag}_\ell^*})\beta^i}\}_{i=0}^{n_3}, T\right)$$

where T is either $m_b \cdot e(g,g)^r$ or a random value $R \in \mathbb{G}_T$ according to the definition of the oracles. To create the challenge cipherttext for $\mathcal{A}$, $\mathcal{B}$ requests $(h_{\mathsf{id}^*||0}, h_{\mathsf{id}^*||\mathsf{tag}_1^*}, \ldots, h_{\mathsf{id}^*||\mathsf{tag}_{\ell-1}^*}$ to the oracle it works with. It then, computes $a_0, \ldots, a_\ell$ where a_i is a coefficient of x^i of $(x + h_{\mathsf{id}^*||0})(x + h_{\mathsf{id}^*||\mathsf{tag}_1^*}) \cdots (x + h_{\mathsf{id}^*||\mathsf{tag}_{\ell-1}^*})$.

$$\begin{aligned}\mathsf{ct} &= \left(\{R_1, \ldots, R_\ell, \{g^{r(\beta+h_{\mathsf{id}^*||\mathsf{tag}_\ell^*})(a_\ell\beta^\ell+\ldots+a_1\beta^1+a_0)\beta^i}\}_{i=0}^{n_3-\ell}\}, T\right)\\ &= \left(\{R_1, \ldots, R_\ell, \{g^{r(\beta+h_{\mathsf{id}^*||\mathsf{tag}_\ell^*})(\beta+h_{\mathsf{id}^*||0})(\beta+h_{\mathsf{id}^*||\mathsf{tag}_1^*})\cdots(\beta+h_{\mathsf{id}^*||\mathsf{tag}_{\ell-1}^*})\beta^i}\}_{i=0}^{n_3-\ell}\}, T\right)\end{aligned}$$

where $R_1, \ldots, R_\ell$ are random values in $\mathbb{Z}_p$ It, then, returns the challenge chiphertext to $\mathcal{A}$. The last equality of the above equation because, for all $i = \{0, 1, \ldots, \ell\}$, a_i is also the coefficient of β^i of $(\beta + h_{\mathsf{id}^*||0})(\beta + h_{\mathsf{id}^*||\mathsf{tag}_1^*}) \cdots (\beta + h_{\mathsf{id}^*||\mathsf{tag}_{\ell-1}^*})$ by its definition.

Guess: When $\mathcal{B}$ receives the answer from $\mathcal{A}$, it sends back the result to the oracle it works with. As $\mathcal{A}$ distinguishes if T encrypts a random message or m_b with the non-negligible advantage ϵ, $\mathcal{B}$ also can use this advantage to distinguish between $\mathcal{O}^0_{AIBE}$ and $\mathcal{O}^1_{AIBE}$. □

5.3. Performance Evaluation

In this section, we compare the performance of our scheme with the other revocable identity-based encryption schemes in Tables 2 and 3. In revocable identity-based encryption schemes [9,17,26], backward compatibility is not supported. Therefore, we assume that they keep all previous keys for backward compatibility for n time periods.

As shown in those tables, our scheme has a very short private key (sk) even though it supports backward compatibility. In the existing scheme, the size of those keys increases in not only the number of keys it keeps for backward compatibility, (n) but also the total number of users (ℓ). Moreover, our key update processes are simple, it always needs a single key for each time period and sk is directly used as a decryption key. In all other schemes, the size of updating keys increases in the total number of users (ℓ).

Our scheme has relatively longer ciphertexts as it increases in the maximum number of updates and also the decryption needs n exponentiation computations although it only needs one pairing. However, the other RIBE does not support the updates. It needs decryption and re-encryption to update existing ciphertexts. This difference makes the size of the ciphertext larger but we believe that this is the cost of the functional enhancement. In addition, one may consider guaranteeing backward compatibility within reasonable time periods. With a smaller n, our scheme outperforms the other RIBE schemes in terms of decryption overhead.

Moreover, the size of the secret keys is often considered an expensive resource as it needs secure memory and is an extra burden for the management. Therefore, our proposed scheme can be used where the restriction of secure memory is severe.

Table 2. Efficiency Comparison with RIBE-parameter sizes. (n is the maximum number of backward-compatible ciphertexts. ℓ is the maximum number of users.)

	pk	**sk with BC**	**ct**
SE [9]	$(6+\|ID\|)\|\mathbb{G}_p\|$	$2n(log\ell)\|\mathbb{G}_p\|$	$3\|\mathbb{G}_p\|+\|\mathbb{G}_T\|$
LLP [17]	$6\|\mathbb{G}_N\|+\|\mathbb{G}_T\|$	$n(log^{1.5}\ell)\|\mathbb{G}_N\|$	$4\|\mathbb{G}_N\|$
ESW [26]	$7\|\mathbb{G}_1\|+11\|\mathbb{G}_2\|+\|\mathbb{Z}_p\|$	$n(5log\ell)\|\mathbb{G}_2\|$	$4\|\mathbb{G}_1\|+\|\mathbb{G}_T\|+\|\mathbb{Z}_p\|$
Ours	$(n+2)\|\mathbb{G}_p\|+\|\mathbb{Z}_p\|$	$\|\mathbb{G}_p\|$	$(n+1)\|\mathbb{G}_p\|+\|\mathbb{G}_T\|$

Table 3. Efficiency Comparison with RIBE - Updating keys and # of pairing in Decryption (r is the number of revoked users. ℓ is the maximum number of users.

	Update Key	**Decryption Key**	**# of Pairing in Dec**
SE [9]	$(2rlog(\ell/r))\|\mathbb{G}_p\|$	$3\|\mathbb{G}_p\|$	3 Pairings
LLP [17]	$4r\|\mathbb{G}_N\|$	$3\|\mathbb{G}_N\|$	3 Pairings + 10 Exp.
ESW [26]	$3rlog(\ell/r))\|\mathbb{G}_2\|$	$6\|\mathbb{G}_2\|$	3 Pairings + 2 Exp.
Ours	$r\|\mathbb{G}_p\|$	$\|\mathbb{G}_p\|$	1 Pairing + r Exp.

6. Discussion and Conclusions

6.1. Discussion on Threats on Updating Ciphertexts

In our proposed scheme, the ciphertext update can be conducted by any third party without giving any private key. However, our update only makes the access policy applied to ciphertexts more restrictive because it always needs a newly established update key to be aggregated to a decryption key in addition to the keys already aggregated to the decryption key. Due to this, even one who has a malicious purpose cannot compromise the ciphertext through the update. The other potential threat is one makes ciphertexts inaccessible by updating them to an arbitrary identity. However, this means that the adversary has a writing privilege for the ciphertext because the update process needs to overwrite the previous ciphertext to the updated one. If the adversary already has the writing privilege, the adversary can compromise the availability of ciphertext anyway even without using the update algorithm. For example, it can overwrite ciphertext to any random elements using its privilege. Due to the reasons we explain above, we argue that the threats on updating ciphertext are reasonable and do not increase the attacker's capability significantly.

6.2. Conclusions

In this paper, we introduce a new encryption system, which supports the backward compatibility property. Our proposed scheme allows for a user to keep a single key allocated for the present time slot and this key can decrypt all ciphertexts created at past and present time slots. So, it is backward compatible. In addition, in our proposed scheme, ciphertexts can be updated for the new time slots without decryption. After it is updated, it cannot be decrypted using the past keys. Therefore, it naturally supports revocation. The ciphertext update process does not require any secret parameters in our scheme so that it can be delegated to a third party. We believe that this is helpful for cloud storage and server-aided IoT network where the synchronization of the data in a system matters.

We present our idea by setting a new definition of backward compatible identity-based encryption (BC-IBE) and its security model. Furthermore, we construct an efficient scheme that satisfies the backward compatibility property we previously described. We prove its security using the aggregate identity-based encryption scheme introduced by Goyal and Vaikuntanathan.

We provide an efficient scheme as the size of the private key is constant, but the length of the ciphertext of the proposed scheme increases linearly over the maximum number of updates. It would be interesting to reduce the size of ciphertexts for future work.

Funding: This work was supported by the Ewha Womans University Research Grant of 2022.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data sharing not applicable.

Conflicts of Interest: The author declares no conflict of interest.

Appendix A. Goyal and Vaikuntanathan's Aggregate IBE

We provide the construction of Goyal and Vaikuntanathan's Aggregate IBE [15]. We let the bilinear group $\langle p, \mathbb{G}, \mathbb{G}_T, g, e\rangle \leftarrow \mathcal{G}(1^\lambda)$ be the bilinear group parameters in prime order p.

- $\mathsf{Setup}(1^\lambda, 1^n) \rightarrow (\mathsf{pk}, \mathsf{msk})$ takes the security parameter λ as input and the upper bound on the number of aggregations n. It randomly samples a value $\beta \in \mathbb{Z}_p^*$ and the public parameters for identity hashing as $\mathsf{hk} \leftarrow \mathsf{HGen}(1^\lambda)$. It sets a public key $\mathsf{pk} := (\mathsf{hk}, \{g^{\beta^i}\}_{i\in[n]}, g^\alpha)$ and a private key $\mathsf{msk} := (\mathsf{hk}, \beta)$.
- $\mathsf{KeyGen}(\mathsf{id}, \mathsf{msk}) \rightarrow \mathsf{sk}_{\mathsf{id}}$: The key generation algorithm takes the identity id and the master secret key msk. It hashes the identity as $h_{\mathsf{id}} = H(\mathsf{hk}, \mathsf{id})$. It outputs the secret key $\mathsf{sk}_{\mathsf{id}}$ as
$$g^{1/(\beta+h_{\mathsf{id}})}.$$
- $\mathsf{Aggregate}(\mathsf{pk}, \{(\mathsf{id}_i, sk_i)\}_i) \rightarrow \hat{\mathsf{sk}}$: The key aggregation algorithm computes the aggregated key as
$$\hat{\mathsf{sk}} = \mathsf{DPP}(\{\mathsf{sk}_i, x_i\}_i)$$
where $x_i = H(\mathsf{hk}, \mathsf{id}_i)$.
- $\mathsf{Enc}(m, \mathsf{mpk}, \mathsf{id}, 1^T) \rightarrow \mathsf{ct}$: The encryption algorithm takes a message m, a public key pk, an identity id and a bound T where T is the maximum number of aggregations. It randomly selects a random value $r \in \mathbb{Z}_p$ and computes $h_{\mathsf{id}} = H(\mathsf{hk}, \mathsf{id})$. It, then, sets the following as the ciphertext ct:
$$C_i := g^{r(\beta+h_{\mathsf{id}})\beta^i}, C_T := m \cdot e(g,g)^r$$
It outputs $\mathsf{ct} := (\{C_i\}_{i=0}^{T-1}, C_T)$.
- $\mathsf{AggDec}(\hat{\mathsf{sk}}, \mathsf{ct}, (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), j \in [\ell]) \rightarrow m$: The decryption algorithm takes the aggregated secret key $\hat{\mathsf{sk}}$ and the ciphertext ct and the identities $\{\mathsf{id}_1, \ldots \mathsf{id}_\ell\}$. It parses ct as $(\{A_i\}_{i=0}^{T-1}, B)$ and computes the identity hash $x_i = H(\mathsf{hk}, \mathsf{id}_i)$ for $i \in [\ell] \setminus \{j\}$. It then computes the coefficients $a_i \in \mathbb{Z}_p$ for i $\in \{0, \ldots, \ell-1\}$ as in
$$\prod_{i\in[\ell]\setminus\{j\}} (y + x_i) = \sum_{i=0}^{\ell-1} a_i \cdot y^i (\mathsf{mod}\ \ p).$$
It, then, outputs
$$m = C_T \cdot e(\prod_{i=0}^{\ell-1} {C_i}^{a_i}, \mathsf{sk}_{\mathsf{id},\ell})^{-1}.$$
The correctness of the above scheme can be found in [15].

References

1. Huang, L.; Rice, A.; Ellingsen, E.; Jackson, C. Analyzing Forged SSL Certificates in the Wild. In Proceedings of the 2014 IEEE Symposium on Security and Privacy, SP 2014, Berkeley, CA, USA, 18–21 May 2014; pp. 83–97. [CrossRef]
2. Naylor, D.; Schomp, K.; Varvello, M.; Leontiadis, I.; Blackburn, J.; López, D.R.; Papagiannaki, K.; Rodríguez, P.R.; Steenkiste, P. Multi-Context TLS (mcTLS): Enabling Secure In-Network Functionality in TLS. In Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication, SIGCOMM 2015, London, UK, 17–21 August 2015; Uhlig, S., Maennel, O., Karp, B., Padhye, J., Eds.; ACM: New York, NY, USA, 2015; pp. 199–212. [CrossRef]
3. Shamir, A. Identity-Based Cryptosystems and Signature Schemes. In *Advances in Cryptology, Proceedings of CRYPTO*; Lecture Notes in Computer Science; Blakley, G.R., Chaum, D., Eds.; Springer: Berlin/Heidelberg, Germany, 1984; Volume 196, pp. 47–53.
4. Boneh, D.; Franklin, M.K. Identity-Based Encryption from the Weil Pairing. In Proceedings of the Advances in Cryptology-CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, CA, USA, 19–23 August 2001; Lecture Notes in Computer Science; Kilian, J., Ed.; Springer: Berlin/Heidelberg, Germany, 2001; Volume 2139, pp. 213–229. [CrossRef]
5. Waters, B. Dual System Encryption: Realizing Fully Secure IBE and HIBE under Simple Assumptions. In Proceedings of the Advances in Cryptology-CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, 16–20 August 2009; Lecture Notes in Computer Science; Halevi, S., Ed.; Springer: Berlin/Heidelberg, Germany, 2009; Volume 5677, pp. 619–636. [CrossRef]
6. Ma, X.; Lin, D. Generic Constructions of Revocable Identity-Based Encryption. In Proceedings of the Information Security and Cryptology—15th International Conference, Inscrypt 2019, Nanjing, China, 6–8 December 2019; Revised Selected Papers; Lecture Notes in Computer Science; Liu, Z., Yung, M., Eds.; Springer: Berlin/Heidelberg, Germany, 2019; Volume 12020, pp. 381–396. [CrossRef]
7. Katsumata, S.; Matsuda, T.; Takayasu, A. Lattice-Based Revocable (Hierarchical) IBE with Decryption Key Exposure Resistance. In Proceedings of the Public-Key Cryptography-PKC 2019-22nd IACR International Conference on Practice and Theory of Public-Key Cryptography, Beijing, China, 14–17 April 2019; Proceedings, Part II; Lecture Notes in Computer Science; Lin, D., Sako, K., Eds.; Springer: Berlin/Heidelberg, Germany, 2019; Volume 11443, pp. 441–471. [CrossRef]
8. Emura, K.; Takayasu, A.; Watanabe, Y. Generic Constructions of Revocable Hierarchical Identity-based Encryption. *IACR Cryptol. ePrint Arch.* **2021**, 515.
9. Seo, J.H.; Emura, K. Revocable Identity-Based Cryptosystem Revisited: Security Models and Constructions. *IEEE Trans. Inf. Forensics Secur.* **2014**, *9*, 1193–1205. [CrossRef]
10. Takayasu, A.; Watanabe, Y. Revocable identity-based encryption with bounded decryption key exposure resistance: Lattice-based construction and more. *Theor. Comput. Sci.* **2021**, *849*, 64–98. [CrossRef]
11. Boneh, D.; Boyen, X.; Goh, E.J. Hierarchical Identity Based Encryption with Constant Size Ciphertext. In *Advances in Cryptology–EUROCRYPT 2005: 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, 22–26 May 2005*; Springer: Berlin/Heidelberg, Germany, 2005; pp. 440–456.
12. Boneh, D.; Boyen, X. Efficient Selective-ID Secure Identity-Based Encryption Without Random Oracles. In *Advances in Cryptology-EUROCRYPT 2004: International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, 2–6 May 2004*; Springer: Berlin/Heidelberg, Germany, 2004; pp. 223–238.
13. Boldyreva, A.; Goyal, V.; Kumar, V. Identity-based encryption with efficient revocation. In Proceedings of the 2008 ACM Conference on Computer and Communications Security, CCS 2008, Alexandria, VA, USA, 27–31 October 2008; Ning, P., Syverson, P.F., Jha, S., Eds.; ACM: New York, NY, USA, 2008; pp. 417–426. [CrossRef]
14. Libert, B.; Vergnaud, D. Adaptive-ID Secure Revocable Identity-Based Encryption. In Proceedings of the Topics in Cryptology-CT-RSA 2009, The Cryptographers' Track at the RSA Conference 2009, San Francisco, CA, USA, 20–24 April 2009; Lecture Notes in Computer Science; Fischlin, M., Ed.; Springer: Berlin/Heidelberg, Germany, 2009; Volume 5473, pp. 1–15. [CrossRef]
15. Goyal, R.; Vaikuntanathan, V. Locally Verifiable Signature and Key Aggregation. In Proceedings of the Advances in Cryptology-CRYPTO 2022—42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, 15–18 August 2022; Proceedings, Part II; Lecture Notes in Computer Science; Dodis, Y., Shrimpton, T., Eds.; Springer: Berlin/Heidelberg, Germany, 2022; Volume 13508, pp. 761–791. [CrossRef]
16. Chen, J.; Lim, H.W.; Ling, S.; Wang, H.; Nguyen, K. Revocable Identity-Based Encryption from Lattices. In Proceedings of the Information Security and Privacy—17th Australasian Conference, ACISP 2012, Wollongong, NSW, Australia, 9–11 July 2012; Lecture Notes in Computer Science; Susilo, W., Mu, Y., Seberry, J., Eds.; Springer: Berlin/Heidelberg, Germany, 2012; Volume 7372, pp. 390–403. [CrossRef]
17. Lee, K.; Lee, D.H.; Park, J.H. Efficient revocable identity-based encryption via subset difference methods. *Des. Codes Cryptogr.* **2017**, *85*, 39–76. [CrossRef]
18. Zhang, Y.; Liu, X.; Hu, Y. Simplified Server-Aided Revocable Identity-Based Encryption from Lattices. In Proceedings of the Provable and Practical Security—16th International Conference, ProvSec 2022, Nanjing, China, 11–12 November 2022; Lecture Notes in Computer Science; Ge, C., Guo, F., Eds.; Springer: Berlin/Heidelberg, Germany, 2022; Volume 13600, pp. 71–87. [CrossRef]

19. Qin, B.; Deng, R.H.; Li, Y.; Liu, S. Server-Aided Revocable Identity-Based Encryption. In Proceedings of the Computer Security-ESORICS 2015-20th European Symposium on Research in Computer Security, Vienna, Austria, 21–25 September 2015; Proceedings, Part I; Lecture Notes in Computer Science; Pernul, G., Ryan, P.Y.A., Weippl, E.R., Eds.; Springer: Berlin/Heidelberg, Germany, 2015; Volume 9326, pp. 286–304. [CrossRef]
20. Nguyen, K.; Wang, H.; Zhang, J. Server-Aided Revocable Identity-Based Encryption from Lattices. In Proceedings of the Cryptology and Network Security—15th International Conference, CANS 2016, Milan, Italy, 14–16 November 2016; Lecture Notes in Computer Science; Foresti, S.; Persiano, G., Eds.; Springer: Berlin/Heidelberg, Germany, 2016; Volume 10052, pp. 107–123. [CrossRef]
21. Sahai, A.; Seyalioglu, H.; Waters, B. Dynamic Credentials and Ciphertext Delegation for Attribute-Based Encryption. In Proceedings of the Advances in Cryptology-CRYPTO 2012—32nd Annual Cryptology Conference, Santa Barbara, CA, USA, 19–23 August 2012; Lecture Notes in Computer Science; Safavi-Naini, R., Canetti, R., Eds.; Springer: Berlin/Heidelberg, Germany, 2012; Volume 7417, pp. 199–217. [CrossRef]
22. Kim, J.; Susilo, W.; Baek, J.; Nepal, S.; Liu, D. Ciphertext-Delegatable CP-ABE for a Dynamic Credential: A Modular Approach. In Proceedings of the Information Security and Privacy—24th Australasian Conference, ACISP 2019, Christchurch, New Zealand, 3–5 July 2019; Lecture Notes in Computer Science; Jang-Jaccard, J., Guo, F., Eds.; Springer: Berlin/Heidelberg, Germany, 2019; Volume 11547, pp. 3–20. [CrossRef]
23. Ge, C.; Susilo, W.; Baek, J.; Liu, Z.; Xia, J.; Fang, L. Revocable Attribute-Based Encryption with Data Integrity in Clouds. *IEEE Trans. Depend. Secur. Comput.* **2022**, *19*, 2864–2872. [CrossRef]
24. Lee, K.; Park, S. Revocable hierarchical identity-based encryption with shorter private keys and update keys. *Des. Codes Cryptogr.* **2018**, *86*, 2407–2440. [CrossRef]
25. Sun, Y.; Chatterjee, P.; Chen, Y.; Zhang, Y. Efficient Identity-Based Encryption with Revocation for Data Privacy in Internet of Things. *IEEE Internet Things J.* **2022**, *9*, 2734–2743. [CrossRef]
26. Emura, K.; Seo, J.H.; Watanabe, Y. Efficient revocable identity-based encryption with short public parameters. *Theor. Comput. Sci.* **2021**, *863*, 127–155. [CrossRef]
27. Delerablée, C. Identity-Based Broadcast Encryption with Constant Size Ciphertexts and Private Keys. In *Advances in Cryptology–ASIACRYPT 2007: 13th International Conference on the Theory and Application of Cryptology and Information Security, Kuching, Malaysia, 2–6 December 2007*; Lecture Notes in Computer Science; Kurosawa, K., Ed.; Springer: Berlin/Heidelberg, Germany, 2007; Volume 4833, pp. 200–215.
28. Gentry, C.; Waters, B. Adaptive Security in Broadcast Encryption Systems (with Short Ciphertexts). In *Advances in Cryptology-EUROCRYPT 2009: 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, 26–30 April 2009*; Lecture Notes in Computer Science; Joux, A., Ed.; Springer: Berlin/Heidelberg, Germany, 2009; Volume 5479, pp. 171–188.
29. Kim, J.; Camtepe, S.; Susilo, W.; Nepal, S.; Baek, J. Identity-Based Broadcast Encryption with Outsourced Partial Decryption for Hybrid Security Models in Edge Computing. In Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security, AsiaCCS 2019, Auckland, New Zealand, 9–12 July 2019; Galbraith, S.D., Russello, G., Susilo, W., Gollmann, D., Kirda, E., Liang, Z., Eds.; ACM: New York, NY, USA, 2019; pp. 55–66. [CrossRef]
30. Delerablée, C.; Paillier, P.; Pointcheval, D. Fully Collusion Secure Dynamic Broadcast Encryption with Constant-Size Ciphertexts or Decryption Keys. In *Pairing-Based Cryptography–Pairing 2007: First International Conference, Tokyo, Japan, 2–4 July 2007*; Lecture Notes in Computer Science; Takagi, T., Okamoto, T., Okamoto, E., Okamoto, T., Eds.; Springer: Berlin/Heidelberg, Germany, 2007; Volume 4575, pp. 39–59.
31. Delerablée, C.; Pointcheval, D. Dynamic Threshold Public-Key Encryption. In Proceedings of the Advances in Cryptology-CRYPTO 2008, 28th Annual International Cryptology Conference, Santa Barbara, CA, USA, 17–21 August 2008; Lecture Notes in Computer Science; Wagner, D.A., Ed.; Springer: Berlin/Heidelberg, Germany, 2008; Volume 5157, pp. 317–334. [CrossRef]