Outsider-Anonymous Broadcast Encryption with Sublinear Ciphertexts*

Nelly Fazio[1,2] and Irippuge Milinda Perera[2]

[1]The City College of CUNY
fazio@cs.ccny.cuny.edu
[2]The Graduate Center of CUNY
{nfazio,iperera}@gc.cuny.edu

April 10, 2012

Abstract

In the standard setting of broadcast encryption, information about the receivers is transmitted as part of the ciphertext. In several broadcast scenarios, however, the identities of the users authorized to access the content are often as sensitive as the content itself. In this paper, we propose the first broadcast encryption scheme with sublinear ciphertexts to attain meaningful guarantees of receiver anonymity. We formalize the notion of *outsider-anonymous broadcast encryption* (oABE), and describe generic constructions in the standard model that achieve outsider-anonymity under adaptive corruptions in the chosen-plaintext and chosen-ciphertext settings. We also describe two constructions with enhanced decryption, one under the gap Diffie-Hellman assumption, in the random oracle model, and the other under the decisional Diffie-Hellman assumption, in the standard model.

Keywords: Recipient Privacy, Broadcast Encryption, Anonymous IBE, Subset Cover Framework.

1 Introduction

Conventional encryption provides the means for secret transmission of data in point-to-point communication. The setting of broadcast encryption [5,20], instead, consists of a *sender*, an insecure unidirectional *broadcast channel*, and a universe of *receivers*. When the sender wants to transmit some digital content, it specifies the set of authorized receivers and creates an encrypted version of the content. A secure broadcast encryption scheme enables legitimate receivers to recover the original content, while ensuring that excluded users just obtain meaningless data, even in the face of collusions.

The intrinsic access control capabilities of broadcast encryption schemes make them a useful tool for many natural applications, spanning from protecting copyrighted content distributed as stored

media [1], to managing digital subscriptions to satellite TV, to controlling access in encrypted file systems [10]. Thanks to its versatility, broadcast encryption has received a lot of attention from the crypto research community in recent years (see *e.g.*, [9, 11, 16–19, 21, 23, 24, 28]). The quest, however, has been for ever more efficient solutions in terms of broadcast communication, key storage and encryption/decryption running time. Little attention, instead, has been devoted to the exploration of refined security models that accurately account for the requirements inherent in multi-recipient communication. More specifically, the focus has been on providing assurance for sender-oriented properties, while overlooking the security and privacy concerns of the receivers.

One problem with the above (informal) definition of broadcast encryption is the implicit requirement that, whenever the digital content is encrypted and sent in broadcast, information about the set of authorized receivers is necessary to decrypt it correctly. Therefore, the set of authorized receivers is transmitted as part of the ciphertext. This in particular implies that an eavesdropper, even if unable to recover the message, can still easily discover the identities of the actual receivers of the content. A way to address the privacy implications that result from specifying explicitly the set of authorized receivers in the broadcast is to use ephemeral IDs and to keep secret the table that associates such IDs with the actual receivers. This simple solution, however, would at best result in a pseudonym system, in which it is still possible to link pseudonyms across transmissions and determine whether the same entity is an authorized receiver for two different broadcasts.

Anonymous Broadcast Encryption. An interesting variant of the broadcast encryption setting was proposed by Barth *et al.* in [4]. Therein, the authors introduce the notion of *private* broadcast encryption scheme, explicitly aiming to protect the identities of the receivers. As a proof-of-concept, they also suggest both generic and number-theoretic public-key constructions that do not leak any information about the list of authorized receivers, and are secure in the standard model and in the random oracle model, respectively. The proposed schemes, however, have communication complexity linear in the number of recipients. In [27], Libert *et al.* recently suggested proof techniques to argue the security of (a variant of) the number-theoretic construction of [4] without reliance on random oracles, thus attaining anonymous broadcast encryption with efficient decryption in the standard model. Still, ciphertexts in the resulting construction have length linear in the number of recipients.

Krzywiecki *et al.* presented a private public-key broadcast encryption scheme with communication complexity proportional to the number of revoked users [26]. The security analysis of the proposed solution is rather informal, however, so the security guarantees are at best heuristic.

In [29], Yu *et al.* presented the first *secret-key* multicast scheme with membership anonymity and communication complexity independent of the number of receivers. The proposed scheme not only hides the *identities* of the receives, but also *the number* of users allowed to receive the content. A shortcoming is that only a single user can be revoked for each broadcast.

A promising research line toward practical receiver-anonymous broadcast encryption has recently been started by Jarecki and Liu [25]. The authors propose the first construction of an efficient unlinkable secret handshake scheme, which is an authenticated key exchange protocol providing *affiliation/policy hiding* (*i.e.*, the transmission hides the affiliation and the identities of all parties) and *unlinkability* (*i.e.*, it is impossible to link any two instances of the secret handshake protocol). The proposed construction can be seen as a *stateful* version of a public-key broadcast encryption scheme, with the additional property of protecting the receivers' identities. Statefulness, however, implies that the key used to encrypt the broadcasts changes for each transmission, and receivers need to keep track of the changes to be able to recover the content.

An interesting trait of the of construction of [25] is that it trades some degree of anonymity for

Table 1: Comparison of the main efficiency parameters of our oABE schemes with [4] and [27]. Our constructions trade full anonymity (achieved by [4, 27]) for sublinear ciphertexts. The second half shows the schemes with a tagging mechanism allowing only 1 decryption attempt per ciphertext. N is the total number of users in the system. r is the number of revoked users of a ciphertext.

Scheme	MPK Length	sk Length	c Length	Decrypt Attempts
BBW06 [4]	$O(N)$	$O(1)$	$O(N-r)$	$(N-r)/2$
LPQ12 [27]	$O(N)$	$O(1)$	$O(N-r)$	$(N-r)/2$
oABE [Sect. 4.1]	$O(1)$	$O(\log N)$	$O\big(r\log\big(\frac{N}{r}\big)\big)$	$\big(\big\lfloor r\log\big(\frac{N}{r}\big)\big\rfloor \log N\big)/2$
BBW06 [4]	$O(N)$	$O(1)$	$O(N-r)$	1
LPQ12 [27]	$O(N)$	$O(1)$	$O(N-r)$	1
oABE [Sect. 4.4]	$O(N)$	$O(\log N)$	$O\big(r\log\big(\frac{N}{r}\big)\big)$	1
oABE [Sect. 4.5]	$O(N\log N)$	$O(N)$	$O(r)$	1

better efficiency: while the receiver's identities are hidden from outsiders, the scheme still allows authorized users to learn information about other members of the receiver set.

Our Contributions. In this paper we propose the first broadcast encryption scheme with sublinear ciphertexts to achieve meaningful guarantees of receiver anonymity. In particular, we formalize the notion of *outsider-anonymous broadcast encryption* (oABE), and describe a generic construction based on any anonymous identity-based encryption (AIBE) scheme. Compared with the work of [25], our construction has the advantage of being *stateless*, and having constant master public key size.

Additionally, by adapting the techniques of [4], we also obtain an efficient construction with enhanced decryption, where for a given oABE ciphertext, the decryption algorithm executes a single AIBE decryption operation. As outlined in Table 1, by relaxing the anonymity guarantees, we achieve sublinear ciphertexts size in our constructions.

Organization. Sect. 2 provides a brief review of the Subset Cover Framework [28] and of anonymous identity-based encryption [2, 22]. The setting of outsider-anonymous broadcast encryption is introduced in Sect. 3. In Sect. 4 we first present generic constructions in the standard model that achieve outsider-anonymity under adaptive corruptions in the chosen-plaintext (Sect. 4.1) and chosen-ciphertext (Sect. 4.2) settings. Next, we describe a CCA-secure construction with enhanced decryption under the gap Diffie-Hellman assumption in the random oracle model (Sect. 4.3), and also extend it to the standard model (Sect. 4.4), using the twin-DH-based techniques of [13]. In Sect. 4.5 we also present a variant of the scheme in Sect. 4.4 with even shorter ciphertexts, at a price on the other parameters, most notably user storage and decryption complexity. Finally, we outline an optimization for the private-key setting to accommodate storage-constrained systems and attain constant key storage at the Center, while maintaining efficient decryption and logarithmic storage at the receivers (Sect. 4.6).

2 Background

2.1 The Subset Cover Framework

The *subset cover framework* proposed by Naor *et al.* [28] is an environment for defining and analyzing the security of revocation schemes in the private-key setting, where only the Center can broadcast. The main idea of this framework is to define a collection $\mathcal{S}$ of subsets of the universe of users

$\mathcal{U} = \{1, \ldots, N\}$ in the system, and assign each subset $S_j \in \mathcal{S}$ a long-lived key, which is also provided to the users belonging to S_j. When broadcasting a message m, first the Center determines the set of revoked users $\mathcal{R}$, then it finds a set of disjoint subsets $\mathcal{C}$ from the collection $\mathcal{S}$ that "covers" the set $\mathcal{U}\backslash\mathcal{R}$ of receivers, and finally it encrypts the short-lived session key used to encrypt m under all the long-lived keys associated with each subset in $\mathcal{C}$.

In [28], the authors also provide two instantiations of revocation schemes in the subset cover framework, namely the *complete subtree* (CS) method and the *subset difference* (SD) method. In the CS method, the key assignment is information-theoretic but the ciphertext is $O(r \log(\frac{N}{r}))$ long, whereas in the SD method, the ciphertext length is $O(2\,r - 1)$ but the key assignment is computational, where r is the number of revoked users. Although the ciphertext length of the CS method is asymptotically bigger than that of the SD method, we are still interested in the CS method due to its information-theoretic key assignment nature, which seems to be crucial for efficiently preserving the anonymity of the receivers.

Complete Subtree Method. In the CS method as introduced in [28], the N users in the system are represented as the leaves of a full binary tree $\mathcal{T}$. Since this requires N to be a power of 2, dummy users are added to the system in case N is not a power of 2. The collection $\mathcal{S}$ contains all possible complete subtrees of $\mathcal{T}$. More precisely, $\mathcal{S}$ contains a subtree for every node $v_j \in \mathcal{T}$. Since there are $2N - 1$ nodes in $\mathcal{T}$, $|\mathcal{S}| = 2N - 1$.

As for key assignment, every subtree in $\mathcal{S}$ is assigned a long-lived secret key which is also made available to the users (leaves) of the given subtree. Since any user u_i, for $1 \leq i \leq N$, is a member of all the subtrees rooted at each node v_j, for $1 \leq j \leq \log N + 1$, in the path from the root of $\mathcal{T}$ down to u_i, the length of the user secret key is $O(\log N)$.

The ciphertext length in the CS method is $O(r \log(\frac{N}{r}))$ due to the fact that a logarithmic number of subtrees is required to exclude each of the r revoked users (see [28] for further details).

Extension of the CS Method to the Public-Key Setting. As mentioned earlier, the original CS method applies in the private-key setting. Thus, only the Center can broadcast since only it knows all the long-lived keys associated with each subtree in $\mathcal{S}$. In [16], Dodis and Fazio extended the original CS method to the public-key setting by using a two step process.

The first step is a unique assignment of hierarchical identifiers (HID) to the nodes in $\mathcal{T}$ as follows. First, assign the root of $\mathcal{T}$ a special identifier (ID), which we refer to as `Root`. Then, assign each edge of $\mathcal{T}$ with ID $\mathsf{ID} \in \{0, 1\}$ depending on whether the edge connects its parent node to the left or right child. Now, the HID HID_i of any node $v_i \in \mathcal{T}$ can be computed by concatenating all the edge IDs starting from the root of $\mathcal{T}$ down to v_i and then pre-pending the root ID at the front. Since any prefix of the HID HID_i of v_i represents the valid HID of a parent node of v_i, for the simplicity of notation, we denote by $\mathsf{HID}_{i|j}$ the prefix of the hierarchical identifier HID_i of length j.

The second step is to use identity-based encryption, further explained in Sect. 2.2, to encrypt the short-lived session key during broadcast, essentially porting the original CS method to the public-key setting. This allows any user to broadcast a message since the tree structure of the users $\mathcal{T}$ and the HIDs of the roots of the subtrees of $\mathcal{T}$ are publicly known. In this setting, the Center acts as the trusted authority to provide each user with the $\log N + 1$ IBE secret keys of the HIDs of the roots of the subtrees that the user belongs to.

2.2 Anonymous Identity-Based Encryption (AIBE)

Identity-based encryption (IBE), originally proposed by Shamir in [30], is a public key encryption scheme in which the user public key is an arbitrary bit-string and the user secret key is generated by a trusted authority known as the *private-key generator* (PKG) using its master key. The first implementation of this scheme was given in [7] (further implementations can be found in [6, 8, 31] to name a few).

An IBE scheme is called anonymous, formally called anonymous identity-based encryption (AIBE), if an adversary cannot distinguish the identity under which a ciphertext is generated. This notion of anonymity was first introduced in [2]. Subsequent implementations can be found in [12] and [22]. Given below is the formal definition of an AIBE scheme. We refer the reader to [2] for further details including the formal definition of security.

Definition 2.1: An anonymous identity-based encryption scheme, associated with a message space $\mathcal{MSP}$, and a ciphertext space $\mathcal{CSP}$, is a tuple of probabilistic polynomial algorithms $(\mathsf{Init}, \mathsf{Ext}, \mathsf{Enc}, \mathsf{Dec})$ such that:

$(\mathsf{MPK}, \mathsf{MSK}) \leftarrow \mathsf{Init}(1^\lambda)$: The initialization algorithm Init takes as input the security parameter 1^λ, and outputs the master public key MPK and the master secret key MSK of the system.

$sk_{\mathsf{ID}} \leftarrow \mathsf{Ext}(\mathsf{MPK}, \mathsf{MSK}, \mathsf{ID})$: The key extraction algorithm Ext takes as input the master public key MPK, the master secret key MSK, and an identifier $\mathsf{ID} \in \{0,1\}^*$. It outputs the secret key sk_{ID} capable of decrypting ciphertexts intended for the holder of the given identifier ID.

$c \leftarrow \mathsf{Enc}(\mathsf{MPK}, \mathsf{ID}, m)$: The encryption algorithm Enc algorithm takes as input the master public key MPK, an identifier $\mathsf{ID} \in \{0,1\}^*$, and a message $m \in \mathcal{MSP}$. It then outputs a ciphertext $c \in \mathcal{CSP}$.

$m/\perp \coloneqq \mathsf{Dec}(\mathsf{MPK}, sk_{\mathsf{ID}}, c)$: Given the master public key MPK, a secret key sk_{ID}, and a ciphertext $c \in \mathcal{CSP}$, the decryption algorithm Dec either outputs a message $m \in \mathcal{MSP}$ or the failure symbol $\perp$. We assume that Dec is deterministic.

Correctness. For every $\mathsf{ID} \in \{0,1\}^*$ and every $m \in \mathcal{MSP}$, if sk_{ID} is the secret key output by $\mathsf{Ext}(\mathsf{MPK}, \mathsf{MSK}, \mathsf{ID})$, then $\mathsf{Dec}(\mathsf{MPK}, sk_{\mathsf{ID}}, \mathsf{Enc}(\mathsf{MPK}, \mathsf{ID}, m)) = m$. ◊

Weakly Robust AIBE. The *Robust Encryption*, formalized by Abdalla *et al.* [3], requires that it is hard to produce a ciphertext that is valid for two different users. In [3], the authors define two types of robustness, strong and weak. Informally, an AIBE scheme is called *weakly robust*, if any adversary has negligible advantage in producing two identities $\mathsf{ID}_0, \mathsf{ID}_1$ and a message m such that the encryption of m under ID_0 can be decrypted with the private key associated with ID_1 leading to a non-$\perp$ result. In [3], the authors also provide a transformation algorithm which makes possible to obtain a weakly robust AIBE scheme from a regular AIBE one.

3 Outsider-Anonymous Broadcast Encryption (oABE)

3.1 The Setting

Definition 3.1: An outsider-anonymous broadcast encryption scheme, associated with a universe of users $U = \{1, \ldots, N\}$, a message space $\mathcal{MSP}$, and a ciphertext space $\mathcal{CSP}$, is a tuple of probabilistic polynomial algorithms $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ such that:

$(\mathsf{MPK}, \mathsf{MSK}) \leftarrow \mathsf{Setup}(1^\lambda, N)$: The Setup algorithm takes as input the security parameter 1^λ and the number of users in the system N. It outputs the master public key MPK and the master secret key MSK of the system.

$sk_i \leftarrow \mathsf{KeyGen}(\mathsf{MPK}, \mathsf{MSK}, i)$: The key generation algorithm KeyGen takes as input the master public key MPK, the master secret key MSK, and a user $i \in U$. It outputs the secret key sk_i of the user i.

$c \leftarrow \mathsf{Encrypt}(\mathsf{MPK}, S, m)$: The $\mathsf{Encrypt}$ algorithm takes as input the master public key MPK, the set of receivers $S \subseteq U$, and a message $m \in \mathcal{MSP}$. It then outputs a ciphertext $c \in \mathcal{CSP}$.

$m/\perp \coloneqq \mathsf{Decrypt}(\mathsf{MPK}, sk_i, c)$: Given the master public key MPK, a secret key sk_i, and a ciphertext $c \in \mathcal{CSP}$, the $\mathsf{Decrypt}$ algorithm either outputs a message $m \in \mathcal{MSP}$ or the failure symbol $\perp$. We assume that $\mathsf{Decrypt}$ is deterministic.

Correctness. For every $S \subseteq U$, every $i \in S$, and every $m \in \mathcal{MSP}$, if sk_i is the secret key output by $\mathsf{KeyGen}(\mathsf{MPK}, \mathsf{MSK}, i)$ then $\mathsf{Decrypt}(\mathsf{MPK}, sk_i, \mathsf{Encrypt}(\mathsf{MPK}, S, m)) = m$. $\Diamond$

Notice that the decryption algorithm in the above definition does not require the set of recipients S as an input. We stress that this is crucial for providing any level of anonymity in a broadcast encryption scheme.

3.2 The Security Model

Degrees of Anonymity. The degree of recipient-set anonymity captured in our security model, which we call *outsider-anonymity*, lies between the complete lack of protection that characterizes traditional broadcast encryption schemes as introduced in [20, 23], and the full anonymity provided in schemes such as [4, 27]. In an oABE scheme, when the adversary receives a ciphertext of which she is not a legal recipient, she will be unable to learn anything about the identities of the legal recipients (let alone the contents of the ciphertext). Still, for those ciphertexts for which the adversary is in the authorized set of recipients, she might also learn the identities of some the other legal recipients. This seems a natural relaxation, since often the *contents* of the communication already reveals something about the recipient set. At the same time, our new intermediate definition of security might allow the construction of more efficient anonymous broadcast encryption schemes; for example, in Sect. 4 we describe the first broadcast encryption scheme with sublinear ciphertexts that attains some meaningful recipient-set anonymity guarantees.

CCA Security. We now present the security requirements for a broadcast encryption scheme to be *outsider anonymous* against chosen-ciphertext attacks (CCA). First we define the CCA of an oABE scheme as a game, which we term oABE-IND-CCA, played between a probabilistic polynomial time (PPT) adversary $\mathcal{A}$ and a challenger $\mathcal{C}$. The security requirement is that $\mathcal{A}$'s advantage of winning the oABE-IND-CCA game is negligible. The high-level idea of this game is for any two sets of recipients $S_0, S_1 \in U$, $\mathcal{A}$ cannot distinguish between a ciphertext intended for the recipient set S_0 and a ciphertext intended for the recipient set S_1 given the fact that the $\mathcal{A}$ does not possess the secret key of any user in $S_0 \cup S_1$. We require the two sets S_0, S_1 be the same size in order to avoid trivial attacks. The formal definitions follow.

Definition 3.2: The oABE-IND-CCA game defined for an oABE scheme $\Pi = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$, a PPT adversary $\mathcal{A}$, and a challenger $\mathcal{C}$ is as follows:

Setup: $\mathcal{C}$ runs $(\mathsf{MPK}, \mathsf{MSK}) \leftarrow \mathsf{Setup}(1^\lambda, N)$ and gives $\mathcal{A}$ the resulting master public key MPK, keeping the master secret key MSK to itself. $\mathcal{C}$ also initializes the set of revoked users Rev to be empty.

Phase 1: $\mathcal{A}$ adaptively issues queries $q_1, \ldots, q_m$ where each q_i is one of the following:

- Secret-key query i: $\mathcal{A}$ requests the secret key of the user $i \in U$.
  $\mathcal{C}$ runs $sk_i \leftarrow \mathsf{KeyGen}(\mathsf{MPK}, \mathsf{MSK}, i)$ to generate the secret key sk_i of the user i, adds i to Rev, and sends sk_i to $\mathcal{A}$.
- Decryption query (i, c): $\mathcal{A}$ issues a decryption query where $i \in U$ and $c \in \mathcal{CSP}$. First, $\mathcal{C}$ runs $sk_i \leftarrow \mathsf{KeyGen}(\mathsf{MPK}, \mathsf{MSK}, i)$ to generate the secret key sk_i of the user i. Then, it runs $\mathsf{Decrypt}(\mathsf{MPK}, sk_i, c)$ and gives the output to $\mathcal{A}$.

Challenge: $\mathcal{A}$ gives $\mathcal{C}$ two equal length messages $m_0^*, m_1^* \in \mathcal{MSP}$ and two equal length sets of user identities $S_0^*, S_1^* \subseteq U$ with the restriction that $\mathsf{Rev} \cap (S_0^* \cup S_1^*) = \emptyset$. $\mathcal{C}$ picks a random bit $b^* \in \{0, 1\}$, runs $c^* \leftarrow \mathsf{Encrypt}(\mathsf{MPK}, S_{b^*}^*, m_{b^*}^*)$, and sends c^* to $\mathcal{A}$.

Phase 2: $\mathcal{A}$ adaptively issues additional queries $q_{m+1}, \ldots, q_n$ where each q_i is one of the following:

- Secret-key query i such that $i \notin S_0^* \cup S_1^*$.
- Decryption query (i, c) such that, if $i \in S_0^* \cup S_1^*$, then $c \neq c^*$.

In both cases, $\mathcal{C}$ responds as in *Phase 1*.

Guess: $\mathcal{A}$ outputs a guess $b \in \{0, 1\}$ and wins if $b = b^*$.

We refer to such an adversary $\mathcal{A}$ as an oABE-IND-CCA adversary. The advantage of $\mathcal{A}$ winning the above game is defined as,

$$\mathsf{Adv}_{\mathcal{A},\Pi}^{\text{oABE-IND-CCA}} = \left| \Pr[b = b^*] - \tfrac{1}{2} \right|$$

The probability is over the random bits used by the adversary $\mathcal{A}$ and the challenger $\mathcal{C}$. ◊

Definition 3.3: An oABE scheme $\Pi = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ is $(t, q_{sk}, q_d, \epsilon)$-secure if for any t-time oABE-IND-CCA adversary $\mathcal{A}$ making at most q_{sk} chosen secret-key queries and at most q_d chosen decryption queries, we have that $\mathsf{Adv}_{\mathcal{A},\Pi}^{\text{oABE-IND-CCA}} \leq \epsilon$. As a shorthand, we say that Π is $(t, q_{sk}, q_d, \epsilon)$-oABE-CCA-secure. ◊

CPA Security. The chosen plaintext attack (CPA) of an oABE scheme is defined similarly to the oABE-IND-CCA game with the restriction that the adversary is not allowed to issue any decryption queries during *Phase 1* and *Phase 2*. The adversary is still allowed to issue secret-key queries. The CPA security game is termed oABE-IND-CPA.

Definition 3.4: An oABE scheme $\Pi = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ is (t, q_{sk}, ϵ)-oABE-CPA-secure if Π is $(t, q_{sk}, 0, \epsilon)$-oABE-CCA-secure. ◊

Remark 3.1. Our definition of security of an outsider-anonymous broadcast encryption scheme can be easily transformed to a definition of security of a fully anonymous broadcast encryption scheme by changing the restriction in the *Challenge* phase, which is currently $\mathsf{Rev} \cap (S_0^* \cup S_1^*) = \emptyset$, to $\mathsf{Rev} \cap (S_0^* \triangle S_1^*) = \emptyset$.[1]

[1]For any two sets S_0, S_1, their symmetric difference is denoted by $S_0 \triangle S_1$.

4 Our Constructions

We now present our constructions of outsider-anonymous broadcast encryption schemes. In a nutshell, the key point of our constructions is to combine an anonymized version of the public-key extension by Dodis and Fazio [16] of the CS method by Naor *et al.* [28] with a fully secure weakly robust AIBE scheme such as [22]. Notice that our approach can be seen as a *framework* for achieving an oABE scheme by using any weakly robust AIBE scheme as an underlying primitive.

The ciphertext length in all constructions is $O(r \log(\frac{N}{r}))$ times the ciphertext length of the underlying AIBE scheme, and the user secret key length is $O(\log N)$ times the user secret key length of the underlying AIBE scheme, where r is the number of revoked users and N is the total number of users in the system.

We provide two generic public-key constructions: an oABE-CPA-secure construction in Sect. 4.1 and an oABE-CCA-secure construction in Sect. 4.2. The limitation with both of these constructions is that on average, the $\mathsf{Decrypt}$ algorithm attempts $(\lfloor r \log(\frac{N}{r}) \rfloor \log N)/2$ decryption operations of the underlying AIBE scheme. In Sect. 4.3, we present an enhanced oABE-CCA-secure construction in which for a given oABE ciphertext, the $\mathsf{Decrypt}$ algorithm executes a single AIBE decryption operation. A drawback of this construction is that its security can only be proven in the random oracle model. In Sect. 4.4, we present another enhanced oABE-CCA-secure construction whose security can be proven in the standard model. In Sect. 4.5 we present a variant of the scheme in Sect. 4.4 attaining even shorter ciphertexts, at a price on the other parameters, most notably, user storage and decryption complexity. Finally, in Sect. 4.6, we outline an optimization for the private-key setting to attain constant key storage at the Center, while maintaining efficient decryption and logarithmic storage at the receivers.

For the simplicity of exposition, our constructions encrypt the actual message m. The ciphertext length could be further reduced by using a hybrid encryption where m is encrypted using a private-key encryption algorithm with a secret key k, and k is then encrypted using the oABE scheme.

In all constructions, $\mathcal{T}$ denotes the binary tree of N users in the system with respect to the CS method. For simplicity, we assume that $N = 2^n$.

4.1 A Generic oABE-CPA-Secure Public-Key Construction

Given a weakly robust AIBE scheme $\Pi' = (\mathsf{Init}, \mathsf{Ext}, \mathsf{Enc}, \mathsf{Dec})$, we construct an oABE-CPA-secure scheme $\Pi = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ as follows.

$\mathsf{Setup}(1^\lambda, N)$: Obtain $(\mathsf{MPK}', \mathsf{MSK}') \leftarrow \mathsf{Init}(1^\lambda)$. Output MPK and MSK as

$$\mathsf{MPK} \coloneqq (\mathsf{MPK}', N) \qquad \mathsf{MSK} \coloneqq \mathsf{MSK}'.$$

$\mathsf{KeyGen}(\mathsf{MPK}, \mathsf{MSK}, i)$: Let $\mathsf{HID}_i \coloneqq (\mathtt{Root}, \mathsf{ID}_1, \ldots, \mathsf{ID}_n)$ be the hierarchical identifier associated with user i in the binary tree $\mathcal{T}$. For $k \coloneqq 1$ to $n+1$, compute $sk_{i,k} \leftarrow \mathsf{Ext}(\mathsf{MPK}', \mathsf{MSK}', \mathsf{HID}_{i|k})$. Output the secret key sk_i of user i as

$$sk_i \coloneqq (sk_{i,1}, \ldots, sk_{i,n+1}) .$$

$\mathsf{Encrypt}(\mathsf{MPK}, S, m)$: Let Cover be the family of subtrees covering the set of receivers S according to the CS method. For each subtree T_j in Cover, let HID_j be the hierarchical identifier associated with the root of T_j. Let $l \coloneqq |\mathsf{Cover}|$, $r \coloneqq N - |S|$ and $L \coloneqq \lfloor r \log(\frac{N}{r}) \rfloor$.

For $1 \leq j \leq l$, compute $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathsf{HID}_j, m)$. Choose $\widetilde{m} \leftarrow\!\!\$\ \mathcal{MSP}$.
For $l+1 \leq j \leq L$, compute $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathtt{dummy}, \widetilde{m})$, where $\mathtt{dummy}$ is a special identifier used to obtain padding ciphertext components. Output the ciphertext c as

$$c \coloneqq \Big(c_{\pi(1)}, \ldots, c_{\pi(L)}\Big),$$

where $\pi : \{1, \ldots, L\} \rightarrow \{1, \ldots, L\}$ is a random permutation.

Decrypt(MPK, sk_i, c): Parse the secret key sk_i as the tuple $(sk_{i,1}, \ldots, sk_{i,n+1})$ and the ciphertext c as the tuple $(c_1, \ldots, c_L)$.

1. For $k \coloneqq 1$ to $n+1$,
   a. For $j \coloneqq 1$ to L,
      i. Compute $m \coloneqq \mathsf{Dec}(\mathsf{MPK}', sk_{i,k}, c_j)$.
      ii. If $m \neq \perp$, return m. Otherwise, continue to next j.
   b. If $k = n+1$, return $\perp$. Otherwise, continue to next k.

The correctness of this oABE-CPA-secure generic public-key construction follows from the correctness of the underlying AIBE scheme. In Theorem 4.1 (whose proof is provided in App. A.1), we establish the security of this construction based on the security of the underlying AIBE scheme.

Theorem 4.1: *If* $\Pi' = (\mathsf{Init}, \mathsf{Ext}, \mathsf{Enc}, \mathsf{Dec})$ *is* (t, q_{sk}, ϵ)*-AIBE-CPA-secure, then the above construction is* $((t, q_{sk}, 2\,\epsilon\, r \log(\frac{N}{r}))$*-oABE-CPA-secure.* □

Parameters. When the above construction is instantiated with Gentry's Fully Secure AIBE scheme in the CPA setting [22], we obtain the following parameter lengths. Let $\overline{\mathbb{G}}$ and $\overline{\mathbb{G}}_{\mathrm{T}}$ be the two groups with prime order $\overline{q}$ in Gentry's construction. MSK is just one element in $\mathbb{Z}_{\overline{q}}$ and the integer N. MPK is only 3 group elements in $\overline{\mathbb{G}}$. The user secret key consists of $(\log N + 1)$ elements in $\mathbb{Z}_{\overline{q}}$ and $(\log N + 1)$ elements in $\overline{\mathbb{G}}$. The ciphertext consists of $\lfloor r \log(\frac{N}{r}) \rfloor$ elements in $\overline{\mathbb{G}}$ and $2\lfloor r \log(\frac{N}{r}) \rfloor$ elements in $\overline{\mathbb{G}}_{\mathrm{T}}$. Also notice that the Enc algorithm in Gentry's AIBE-CPA-secure scheme does not require any pairing computations since they can be precomputed.

4.2 A Generic oABE-CCA-Secure Public-Key Construction

Given a weakly robust AIBE scheme $\Pi' = (\mathsf{Init}, \mathsf{Ext}, \mathsf{Enc}, \mathsf{Dec})$ and a strongly existentially unforgeable one-time signature scheme $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$, we construct an oABE-CCA-secure scheme $\Pi = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ as follows.

Setup(1^λ, N): Obtain $(\mathsf{MPK}', \mathsf{MSK}') \leftarrow \mathsf{Init}(1^\lambda)$. Output MPK and MSK as

$$\mathsf{MPK} \coloneqq (\mathsf{MPK}', N) \qquad \mathsf{MSK} \coloneqq \mathsf{MSK}'.$$

KeyGen(MPK, MSK, i): Let $\mathsf{HID}_i \coloneqq (\mathtt{Root}, \mathsf{ID}_1, \ldots, \mathsf{ID}_n)$ be the hierarchical identifier associated with user i in the binary tree $\mathcal{T}$. For $k \coloneqq 1$ to $n+1$, compute $sk_{i,k} \leftarrow \mathsf{Ext}(\mathsf{MPK}', \mathsf{MSK}', \mathsf{HID}_{i|k})$. Output the secret key sk_i of user i as

$$sk_i \coloneqq (sk_{i,1}, \ldots, sk_{i,n+1}).$$

Encrypt(MPK, S, m): Generate $(\mathsf{VK}, \mathsf{SK}) \leftarrow \mathsf{Gen}(1^\lambda)$. Let Cover be the family of subtrees covering the set of receivers S according to the CS method. For each subtree T_j in Cover, let HID_j be the hierarchical identifier associated with the root of T_j.
Let $l \coloneqq |\mathsf{Cover}|$, $r \coloneqq N - |S|$ and $L \coloneqq \lfloor r \log(\frac{N}{r}) \rfloor$.
For $1 \leq j \leq l$, compute $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathsf{HID}_j, \mathsf{VK} \| m)$. Let $\widetilde{m}$ be a random string of the same length as $\mathsf{VK} \| m$. For $l+1 \leq j \leq L$, compute $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathtt{dummy}, \widetilde{m})$, where $\mathtt{dummy}$ is a special identifier used to obtain padding ciphertext components. Compute $\widehat{c}$ as

$$\widehat{c} \coloneqq \Big(c_{\pi(1)}, \ldots, c_{\pi(L)}\Big),$$

where $\pi : \{1, \ldots, L\} \to \{1, \ldots, L\}$ is a random permutation.
Generate $\sigma \leftarrow \mathsf{Sign}(\mathsf{SK}, \mathsf{VK} \| \widehat{c})$, and output $c = (\sigma, \mathsf{VK}, \widehat{c})$.

Decrypt(MPK, sk_i, c): Parse the secret key sk_i as the tuple $(sk_{i,1}, \ldots, sk_{i,n+1})$ and the ciphertext c as $(\sigma, \mathsf{VK}, \widehat{c} = (c_1, \ldots, c_L))$.

1. For $k \coloneqq 1$ to $n+1$,
   a. For $j \coloneqq 1$ to L,
      i. Compute $m' \coloneqq \mathsf{Dec}(\mathsf{MPK}', sk_{i,k}, c_j)$.
      ii. If $m' \neq \perp$, parse m' as $\mathsf{VK} \| m$ and return m if $\mathsf{Vrfy}(\mathsf{VK}, \sigma, \mathsf{VK} \| \widehat{c})$. Otherwise, continue to next j.
   b. If $k = n+1$, return $\perp$. Otherwise, continue to next k.

The correctness of this oABE-CCA-secure generic public-key construction follows from the correctness of the underlying signature and AIBE schemes. In Theorem 4.2 (proof given in App. A.2), we establish the security of this construction.

Theorem 4.2: *If* $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ *is* (t, ϵ_1)*-strongly existentially unforgeable and* $\Pi' = (\mathsf{Init}, \mathsf{Ext}, \mathsf{Enc}, \mathsf{Dec})$ *is* $(t, q_{sk}, q_d, \epsilon_2)$*-AIBE-CCA-secure, then the above construction is* $(t, q_{sk}, q_d, 2(\epsilon_1 + \epsilon_2 \, r \log(\frac{N}{r}))$*-oABE-CCA-secure.* □

Parameters. The parameter lengths of the above construction when instantiated with Gentry's Fully Secure AIBE scheme in the CCA setting [22] are as follows. Let $\overline{\mathbb{G}}$ and $\overline{\mathbb{G}}_\mathrm{T}$ be the two groups with prime order $\overline{q}$ in Gentry's construction. MSK is one element in $\mathbb{Z}_{\overline{q}}$ and the integer N. MPK consists of 5 group elements in $\overline{\mathbb{G}}$ and the definition of a hash function $\overline{H}$ from a family of universal one-way hash functions. The user secret key consists of $3(\log N + 1)$ elements in $\mathbb{Z}_{\overline{q}}$ and $3(\log N + 1)$ elements in $\overline{\mathbb{G}}$. The ciphertext consists of $\lfloor r \log(\frac{N}{r}) \rfloor$ elements in $\overline{\mathbb{G}}$ and $3\lfloor r \log(\frac{N}{r}) \rfloor$ elements in $\overline{\mathbb{G}}_\mathrm{T}$. Similar to Gentry's AIBE-CPA-secure construction, the Enc algorithm in the AIBE-CCA-secure construction does not require any pairing computations since they can be precomputed.

4.3 An Enhanced oABE-CCA-Secure Public-Key Construction in the Random Oracle Model

The main limitation of our generic public-key constructions is the running time of the decryption algorithm. As described in the opening paragraphs of Sect. 4, decryption amounts to performing $(\lfloor r \log(\frac{N}{r}) \rfloor \log N)/2$ AIBE decryption attempts on average. The root cause behind this limitation is the decryption process's inability to identify the correct AIBE ciphertext component efficiently.

In this section, we describe an enhancement of our generic public-key construction under the Diffie-Hellman assumption, in the random oracle model. The main idea of this enhancement is to adapt the techniques of [4] to the structure of our ciphertexts and attach a unique tag to each AIBE ciphertext component of a given oABE ciphertext. With this optimization, the $\mathsf{Decrypt}$ algorithm is able to identify the correct AIBE ciphertext component via a linear search through the whole oABE ciphertext components, at which point a single AIBE decryption operation suffices to recover the original plaintext. This yields an asymptotic decryption time of $O(r\log(\frac{N}{r})\log N)$, but in fact this is in a sense an overestimate, since the cost of searching for the correct ciphertext component is much less than carrying out multiple decryption attempts.

Given a weakly robust AIBE scheme $\Pi' = (\mathsf{Init}, \mathsf{Ext}, \mathsf{Enc}, \mathsf{Dec})$ and a strongly existentially unforgeable one-time signature scheme $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$, we construct an oABE-CCA-secure scheme $\Pi = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ with enhanced decryption as follows. In this construction, $\mathbb{G} = \langle g \rangle$ denotes a group with prime order $q > 2^\lambda$ in which CDH is hard and DDH is easy and g is a group generator. $H : \mathbb{G} \to \{0,1\}^\lambda$ is a cryptographic hash function that will be modeled as a random oracle in the security analysis.

$\mathsf{Setup}(1^\lambda, N)$: Obtain $(\mathsf{MPK}', \mathsf{MSK}') \leftarrow \mathsf{Init}(1^\lambda)$. For each node (with the hierarchical identifier HID) in $\mathcal{T}$, draw $a_{\mathsf{HID}} \leftarrow\!\!\$\ \mathbb{Z}_q$, and compute $A_{\mathsf{HID}} \coloneqq g^{a_{\mathsf{HID}}}$. Output MPK and MSK as

$$\mathsf{MPK} \coloneqq (\mathsf{MPK}', N, \mathbb{G}, g, \{A_{\mathsf{HID}}\}_{\mathsf{HID}\in\mathcal{T}}) \qquad \mathsf{MSK} \coloneqq (\mathsf{MSK}', \{a_{\mathsf{HID}}\}_{\mathsf{HID}\in\mathcal{T}})\,.$$

$\mathsf{KeyGen}(\mathsf{MPK}, \mathsf{MSK}, i)$: Let $\mathsf{HID}_i \coloneqq (\mathtt{Root}, \mathsf{ID}_1, \ldots, \mathsf{ID}_n)$ be the hierarchical identifier associated with user i in the binary tree $\mathcal{T}$. For $k \coloneqq 1$ to $n+1$, set $\overline{sk}_{i,k} \coloneqq a_{\mathsf{HID}_{i|k}}$, and compute $sk_{i,k} \leftarrow \mathsf{Ext}(\mathsf{MPK}', \mathsf{MSK}', \mathsf{HID}_{i|k})$. Output the secret key sk_i of user i as

$$sk_i \coloneqq \Big(\Big(\overline{sk}_{i,1}, sk_{i,1}\Big), \ldots, \Big(\overline{sk}_{i,n+1}, sk_{i,n+1}\Big)\Big).$$

$\mathsf{Encrypt}(\mathsf{MPK}, S, m)$: Generate $(\mathsf{VK}, \mathsf{SK}) \leftarrow \mathsf{Gen}(1^\lambda)$. Let Cover be the family of subtrees covering the set of receivers S according to the CS method. For each subtree T_j in Cover, let HID_j be the hierarchical identifier associated with the root of T_j.
Let $l \coloneqq |\mathsf{Cover}|$, $r \coloneqq N - |S|$ and $L \coloneqq \lfloor r\log(\frac{N}{r})\rfloor$. Draw $s \leftarrow\!\!\$\ \mathbb{Z}_q$, and compute $\overline{c}_0 \coloneqq g^s$.
For $1 \le j \le l$, compute $\overline{c}_j \coloneqq H(A^s_{\mathsf{HID}_j})$, $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathsf{HID}_j, \mathsf{VK} \| A^s_{\mathsf{HID}_j} \| m)$.
Let $\widetilde{m}$ be a random string of the same length as $\mathsf{VK}\|\overline{c}_0\|m$. For $l+1 \le j \le L$, set $s_j \leftarrow\!\!\$\ \mathbb{Z}_q$, and compute $\overline{c}_j \coloneqq H(g^{s_j})$, $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathtt{dummy}, \widetilde{m})$, where $\mathtt{dummy}$ is a special identifier used to obtain padding ciphertext components. Compute $\widehat{c}$ as

$$\widehat{c} \coloneqq \Big(\overline{c}_0, \Big(\overline{c}_{\pi(1)}, c_{\pi(1)}\Big), \ldots, \Big(\overline{c}_{\pi(L)}, c_{\pi(L)}\Big)\Big)\,,$$

where $\pi : \{1, \ldots, L\} \to \{1, \ldots, L\}$ is a random permutation. Generate $\sigma \leftarrow \mathsf{Sign}(\mathsf{SK}, \mathsf{VK}\|\widehat{c})$, and output $c \coloneqq (\sigma, \mathsf{VK}, \widehat{c})$.

$\mathsf{Decrypt}(\mathsf{MPK}, sk_i, c)$: Parse the secret key sk_i as the tuple $((\overline{sk}_{i,1}, sk_{i,1}), \ldots, (\overline{sk}_{i,n+1}, sk_{i,n+1}))$ and the ciphertext c as $(\sigma, \mathsf{VK}, \widehat{c} = (\overline{c}_0, (\overline{c}_1, c_1), \ldots, (\overline{c}_L, c_L)))$.

1. For $k \coloneqq 1$ to $n+1$,
   a. Compute $tag_k \coloneqq H(\overline{c}_0^{\overline{sk}_{i,k}})$

2. Check whether $\exists k \in [1, n+1]\ \exists j \in [1, L]$ such that $tag_k = \overline{c}_j$
   a. If suitable k, j exist, compute $m' \coloneqq \mathsf{Dec}(\mathsf{MPK}', sk_{i,k}, c_j)$.
      If m' can be parsed as $\mathsf{VK} \| \overline{c}_0^{\overline{sk}_{i,k}} \| m$ and $\mathsf{Vrfy}(\mathsf{VK}, \sigma, \mathsf{VK} \| \widehat{c})$, return m.
   b. Otherwise, return $\perp$.

Remark 4.1. Notice that the check in Step 2 of the Decrypt algorithm can be performed in expected time $O(n+L) = O(L)$, *e.g.*, using a hash table $\mathcal{H}$ to compute the intersection between $\{tag_k\}_{k \in [1,n+1]}$ and $\{\overline{c}_j\}_{j \in [1,L]}$ as follows.

1. Initialize $\mathcal{H}$ to be empty.
2. For $k \coloneqq 1$ to $n+1$
   a. Insert (tag_k, k) in $\mathcal{H}$.
3. For $j \coloneqq 1$ to L
   a. Look up an entry of the form $(\overline{c}_j, k)$ in $\mathcal{H}$. If found, return k.

Theorem 4.3: *If* $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ *is* (t, ϵ_1)*-strongly existentially unforgeable,* $\Pi' = (\mathsf{Init}, \mathsf{Ext}, \mathsf{Enc}, \mathsf{Dec})$ *is* $(t, q_{sk}, q_d, \epsilon_2)$*-AIBE-CCA-secure, and CDH is* (t, ϵ_3)*-hard in* $\mathbb{G}$ *and DDH is efficiently computable in* $\mathbb{G}$*, then the above construction is* $(t, q_{sk}, q_d, 2(\epsilon_1+\epsilon_2+\epsilon_3)\, r \log(\frac{N}{r}))$*-oABE-CCA-secure, in the random oracle model.* □

The proof of the above theorem is given in App. A.3.

4.4 An Enhanced oABE-CCA-Secure Public-Key Construction in the Standard Model

In this section, we augment the construction in Sect. 4.3 so that its security can be proven in the standard model under the decisional Diffie-Hellman assumption using techniques from [27]. The key ingredient of this modification is the "trapdoor test" of the twin Diffie-Hellman problem [13].

Let $\Pi' = (\mathsf{Init}, \mathsf{Ext}, \mathsf{Enc}, \mathsf{Dec})$ be a weakly robust AIBE scheme and $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ a strongly existentially unforgeable one-time signature scheme. We construct an oABE-CCA-secure scheme $\Pi = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ with enhanced decryption in the standard model as follows. In this construction, $\mathbb{G} = \langle g \rangle$ denotes a group with prime order $q > 2^\lambda$ in which DDH is hard and g is a group generator.

Setup($1^\lambda, N$): Obtain $(\mathsf{MPK}', \mathsf{MSK}') \leftarrow \mathsf{Init}(1^\lambda)$. For each node (with the hierarchical identifier HID) in $\mathcal{T}$, draw $a_{\mathsf{HID}}, b_{\mathsf{HID}}, c_{\mathsf{HID}}, d_{\mathsf{HID}} \leftarrow\$\ \mathbb{Z}_q$, and compute $A_{\mathsf{HID}} \coloneqq g^{a_{\mathsf{HID}}}, B_{\mathsf{HID}} \coloneqq g^{b_{\mathsf{HID}}}$, $C_{\mathsf{HID}} \coloneqq g^{c_{\mathsf{HID}}}, D_{\mathsf{HID}} \coloneqq g^{d_{\mathsf{HID}}}$. Output MPK and MSK as

$$\mathsf{MPK} \coloneqq (\mathsf{MPK}', N, \mathbb{G}, g, \{A_{\mathsf{HID}}, B_{\mathsf{HID}}, C_{\mathsf{HID}}, D_{\mathsf{HID}}\}_{\mathsf{HID} \in \mathcal{T}})$$
$$\mathsf{MSK} \coloneqq (\mathsf{MSK}', \{a_{\mathsf{HID}}, b_{\mathsf{HID}}, c_{\mathsf{HID}}, d_{\mathsf{HID}}\}_{\mathsf{HID} \in \mathcal{T}}).$$

KeyGen(MPK, MSK, i): Let $\mathsf{HID}_i \coloneqq (\mathtt{Root}, \mathsf{ID}_1, \ldots, \mathsf{ID}_n)$ be the hierarchical identifier associated with user i in the binary tree $\mathcal{T}$. For $k \coloneqq 1$ to $n+1$, set $\overline{sk}_{i,k} \coloneqq (a_{\mathsf{HID}_{i|k}}, b_{\mathsf{HID}_{i|k}}, c_{\mathsf{HID}_{i|k}}, d_{\mathsf{HID}_{i|k}})$, and compute $sk_{i,k} \leftarrow \mathsf{Ext}(\mathsf{MPK}', \mathsf{MSK}', \mathsf{HID}_{i|k})$. Output the secret key sk_i of user i as

$$sk_i \coloneqq \Big(\big(\overline{sk}_{i,1}, sk_{i,1}\big), \ldots, \big(\overline{sk}_{i,n+1}, sk_{i,n+1}\big)\Big).$$

Encrypt(MPK, S, m): Generate $(\mathsf{VK}, \mathsf{SK}) \leftarrow \mathsf{Gen}(1^\lambda)$. Let Cover be the family of subtrees covering the set of receivers S according to the CS method. For each subtree T_j in Cover, let HID_j be the hierarchical identifier associated with the root of T_j.
Let $l \coloneqq |\mathsf{Cover}|$, $r \coloneqq N - |S|$ and $L \coloneqq \lfloor r \log(\frac{N}{r}) \rfloor$. Draw $s \leftarrow\$ \mathbb{Z}_q$, and compute $\overline{c}_0 \coloneqq g^s$.
For $1 \le j \le l$, compute[2] $\overline{c}_j \coloneqq ((A^{\mathsf{VK}}_{\mathsf{HID}_j} B_{\mathsf{HID}_j})^s, (C^{\mathsf{VK}}_{\mathsf{HID}_j} D_{\mathsf{HID}_j})^s)$, $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathsf{HID}_j, \mathsf{VK} \| m)$.
Let $\widetilde{m}$ be a random string of the same length as $\mathsf{VK}\|m$. For $l+1 \le j \le L$, set $s_{j,1}, s_{j,2} \leftarrow\$ \mathbb{Z}_q$, and compute $\overline{c}_j \coloneqq (g^{s_{j,1}}, g^{s_{j,2}})$, $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathtt{dummy}, \widetilde{m})$, where $\mathtt{dummy}$ is a special identifier used to obtain padding ciphertext components. Compute $\widehat{c}$ as

$$\widehat{c} \coloneqq \Big(\overline{c}_0, \Big(\overline{c}_{\pi(1)}, c_{\pi(1)}\Big), \ldots, \Big(\overline{c}_{\pi(L)}, c_{\pi(L)}\Big)\Big),$$

where $\pi : \{1, \ldots, L\} \to \{1, \ldots, L\}$ is a random permutation. Generate $\sigma \leftarrow \mathsf{Sign}(\mathsf{SK}, \mathsf{VK} \| \widehat{c})$, and output $c \coloneqq (\sigma, \mathsf{VK}, c)$.

Decrypt(MPK, sk_i, c): Parse the secret key sk_i as the tuple $((\overline{sk}_{i,1}, sk_{i,1}), \ldots, (\overline{sk}_{i,n+1}, sk_{i,n+1}))$ and the ciphertext c as $(\sigma, \mathsf{VK}, \widehat{c} = (\overline{c}_0, (\overline{c}_1, c_1), \ldots, (\overline{c}_L, c_L)))$.

1. For $k \coloneqq 1$ to $n+1$,
   a. Parse $\overline{sk}_{i,k}$ as (a_k, b_k, c_k, d_k)
   b. Compute $tag_k \coloneqq (\overline{c}_0^{a_k \mathsf{VK}}\, \overline{c}_0^{b_k}, \overline{c}_0^{c_k \mathsf{VK}}\, \overline{c}_0^{d_k})$
2. Check whether $\exists k \in [1, n+1]\ \exists j \in [1, L]$ such that $tag_k = \overline{c}_j$
   a. If suitable k, j exist, compute $m' \coloneqq \mathsf{Dec}(\mathsf{MPK}', sk_{i,k}, c_j)$.
   If m' can be parsed as $\mathsf{VK}\|m$ and $\mathsf{Vrfy}(\mathsf{VK}, \sigma, \widehat{c})$, return m.
   b. Otherwise, return $\perp$.

Remark 4.2. Notice that using a technique similar to the one given in Remark 4.1, we can reduce the tag-searching time in Step 2 of the $\mathsf{Decrypt}$ algorithm from $O(nL)$ to $O(n + L) = O(L)$.

Theorem 4.4: *If* $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ *is* (t, ϵ_1)*-strongly existentially unforgeable,* $\Pi' = (\mathsf{Init}, \mathsf{Ext}, \mathsf{Enc}, \mathsf{Dec})$ *is* $(t, q_{sk}, q_d, \epsilon_2)$*-AIBE-CCA-secure, and DDH is* (t, ϵ_3)*-hard in* $\mathbb{G}$*, then the above construction is* $(t, q_{sk}, q_d, 2(\epsilon_1 + \epsilon_2 + 2(\epsilon_3 + \frac{q_d}{2^\lambda}))r \log(\frac{N}{r}))$*-oABE-CCA-secure.* □

The proof of the above theorem is given in App. A.4.

4.5 An Enhanced oABE-CCA-Secure Public-Key Construction with Shorter Ciphertexts

Below we sketch a variation of our techniques from Sect. 4.4 that results in an oABE scheme with ciphertext length $O(r)$. Unfortunately, this very compact ciphertext length comes at a price on the other parameters.

The idea is to combine the Dodis-Fazio [16] public-key extension of the SD method by Naor *et al.* [28] with a fully secure, weakly robust, *anonymous hierarchical identity-based encryption* (AHIBE) scheme with constant ciphertext length such as [14,15]. Following this approach, we would get the following efficiency parameters.

[2] We assume for simplicity that the verification keys for the underlying one-time signature scheme can be encoded into $\mathbb{Z}_q$, but one can always use UOWHFs otherwise.

Ciphertext Length: $O(r)$ AHIBE ciphertexts

Public Key Length: $O(N \log N)$ public tag components

Secret Key Length: $O(\log^2 N)$ AHIBE secret keys and $O(N)$ secret tag components

Decryption Time: $O(N)$ tag computation/searching time and one AHIBE decryption attempt

4.6 An Enhanced oABE-CCA-Secure Private-Key Construction

The enhanced oABE-CCA-secure public key constructions achieve a major performance gain in the Decrypt algorithm compared to the generic oABE-CCA-secure construction, but it also changes the length of the master public key from $O(1)$ to $O(N)$. This increase in master public key length may not be a concern for many practical constructions, since the master public key can be stored as a static data file on a server on the Internet and also in users' computers. Still, for the private-key setting it is possible to accommodate storage-sensitive systems and attain constant key storage at the Center, while maintaining efficient decryption and logarithmic storage at the receivers.

In particular, recall from Sect. 2.1 that in the private-key setting, only the Center can broadcast messages to the receivers. Thus, the $O(N)$ information from which the tags for efficient decryption are created does not need to be published. Therefore, this information can be compressed into $O(1)$ key storage using a standard trick based on any length-tripling pseudo-random number generator G (*cf. e.g.*, the SD method of Naor *et al.* [28]). In other words, the random exponents associated with the subtrees of $\mathcal{T}$ (*cf.* Sect. 4.3) are now pseudorandomly generated from a single seed, by repeated invocations of G on the left or right third of the result of the previous iteration, based on the path to the root of the subtree at hand. Finally, upon reaching the subtree root, the middle third of the pseudorandom output is used to generate the required exponent.

5 Conclusions and Future Work

In this work, we introduced the notion of outsider-anonymity in the broadcast encryption setting and showed that it enables efficient constructions of broadcast encryption schemes with sublinear communication complexity and meaningful anonymity guarantees. It remains an interesting open problem to construct receiver-anonymous broadcast encryption schemes that at once afford full anonymity to the receivers and attain performance levels comparable to those of standard broadcast encryption systems.

Acknowledgments

Nelly Fazio's research was sponsored in part by NSF award #1117675 and by the U.S. Army Research Laboratory and the U.K. Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon. This

project was also partially supported by PSC-CUNY Awards 63356-00 41 and 64578-00 42, jointly funded by the Professional Staff Congress and the City University of New York.

References

[1] AACS. Advanced access content system. `http://www.aacsla.com/`.

[2] M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee, G. Neven, P. Paillier, and H. Shi. Searchable encryption revisited: Consistency properties, relation to Anonymous IBE, and extensions. In *Advances in Cryptology—CRYPTO*, pages 205–222, 2005.

[3] M. Abdalla, M. Bellare, and G. Neven. Robust encryption. In *Theory of Cryptography—TCC*, pages 480–497, 2010.

[4] A. Barth, D. Boneh, and B. Waters. Privacy in encrypted content distribution using private broadcast encryption. In *Financial Cryptography and Data Security—FC*, pages 52–64, 2006.

[5] S. Berkovits. How to broadcast a secret. In *Advances in Cryptology—EUROCRYPT*, pages 535–541, 1991.

[6] D. Boneh and X. Boyen. Secure identity based encryption without random oracles. In *Advances in Cryptology—CRYPTO*, pages 443–459, 2004.

[7] D. Boneh and M. K. Franklin. Identity-based encryption from the weil pairing. In *Advances in Cryptology—CRYPTO*, pages 213–229, 2001.

[8] D. Boneh, C. Gentry, and M. Hamburg. Space-efficient identity based encryption without pairings. In *IEEE Symposium on Foundations of Computer Science—FOCS*, pages 647–657, 2007.

[9] D. Boneh, C. Gentry, and B. Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In *Advances in Cryptology—CRYPTO*, pages 258–275, 2005.

[10] D. Boneh, E.-J. Goh, N. Modadugu, and H. Shacham. Sirius: Securing remote untrusted storage. In *ISOC Network and Distributed System Security Symposium—NDSS*, pages 131–145, 2003.

[11] D. Boneh and B. Waters. A fully collusion resistant broadcast, trace, and revoke system. In *ACM Conference on Computer and Communications Security—CCS*, pages 211–220, 2006.

[12] X. Boyen and B. Waters. Anonymous hierarchical identity-based encryption (without random oracles). In *Advances in Cryptology—CRYPTO*, pages 290–307, 2006.

[13] D. Cash, E. Kiltz, and V. Shoup. The twin Diffie-Hellman problem and applications. In *Advances in Cryptology—EUROCRYPT*, pages 127–145, 2008.

[14] Y. Chen, Z. Chen, J. Hu, and S. Luo. New fully secure hierarchical identity-based encryption with constant size ciphertexts. In *Information Security Practice and Experience—ISPEC*, pages 55–70, 2011.

[15] J. H. Cheon and J. H. Seo. Fully secure anonymous hierarchical identity-based encryption with constant size ciphertexts. Cryptology ePrint Archive, Report 2011/021, 2011.

[16] Y. Dodis and N. Fazio. Public-key broadcast encryption for stateless receivers. In *Digital Rights Management—DRM*, pages 61–80, 2002.

[17] Y. Dodis and N. Fazio. Public-key trace and revoke scheme secure against adaptive chosen ciphertext attack. In *Public Key Cryptography—PKC*, pages 100–115, 2003.

[18] Y. Dodis, N. Fazio, A. Kiayias, and M. Yung. Scalable public-key tracing and revoking. In *ACM Symposium on Principles of Distributed Computing—PODC*, pages 190–199, 2003. Invited to the PODC '03 Special Issue of Journal of Distributed Computing.

[19] Y. Dodis, N. Fazio, A. Lysyanskaya, and D. Yao. ID-based encryption for complex hierarchies with applications to forward security and broadcast encryption. In *ACM Conference on Computer and Communications Security—CCS*, pages 354–363, 2004.

[20] A. Fiat and M. Naor. Broadcast encryption. In *Advances in Cryptology—CRYPTO*, pages 480–491, 1993.

[21] J. A. Garay, J. Staddon, and A. Wool. Long-lived broadcast encryption. In *Advances in Cryptology—CRYPTO*, pages 333–352, 2000.

[22] C. Gentry. Practical identity-based encryption without random oracles. In *Advances in Cryptology—EUROCRYPT*, pages 445–464, 2006.

[23] C. Gentry and B. Waters. Adaptive security in broadcast encryption systems (with short ciphertexts). In *Advances in Cryptology—EUROCRYPT*, pages 171–188, 2009.

[24] D. Halevy and A. Shamir. The LSD broadcast encryption scheme. In *Advances in Cryptology—CRYPTO*, pages 47–60, 2002.

[25] S. Jarecki and X. Liu. Unlinkable secret handshakes and key-private group key management schemes. In *Applied Cryptography and Network Security—ACNS*, pages 270–287, 2007.

[26] L. Krzywiecki, P. Kubiak, and M. Kutylowski. A revocation scheme preserving privacy. In *Information Security and Cryptology—Inscrypt*, pages 130–143, 2006.

[27] B. Libert, K. G. Paterson, and E. A. Quaglia. Anonymous broadcast encryption. In *Public Key Cryptography—PKC*, pages 206–224, 2012.

[28] D. Naor, M. Naor, and J. Lotspiech. Revocation and tracing schemes for stateless receivers. In *Advances in Cryptology—CRYPTO*, pages 41–62, 2001.

[29] K. Ren, S. Yu, and W. Lou. Attribute-based on-demand multicast group setup with receiver anonymity. In *Security and Privacy in Communication Networks—SecureComm*, pages 18:1–18:6, 2008.

[30] A. Shamir. Identity-based cryptosystems and signature schemes. In *Advances in Cryptology—CRYPTO*, pages 47–53, 1984.

[31] B. Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In *Advances in Cryptology—CRYPTO*, pages 619–636, 2009.

A Security Proofs

Notation. $U = \{1, \ldots, N\}$ is the universe of users. $\mathcal{T}$ denotes the binary tree of N users in the system with respect to the CS method. Let r be the number of revoked users in the challenge ciphertext and $L \coloneqq \lfloor r \log(\frac{N}{r}) \rfloor$. For $b \in \{0, 1\}$, let S_b^* be the sets of authorized receivers chosen by the adversary in the *Challenge* phase. Cover_b denotes the family of subtrees covering the set S_b^* according to the CS method. Let $l_b \coloneqq |\mathsf{Cover}_b|$. For each subtree T_j^b in Cover_b, let HID_j^b be the hierarchical identifier associated with the root of T_j^b where $1 \leq j \leq l_b$.

A.1 Proof of Theorem 4.1

Proof. We organize our proof as a sequence of games, $\mathrm{Game}_0^0, \ldots, \mathrm{Game}_{l_0}^0 \equiv \mathrm{Game}_{l_1}^1, \ldots, \mathrm{Game}_0^1$, between the adversary $\mathcal{A}$ and the challenger $\mathcal{C}$. In the first game (Game_0^0), $\mathcal{A}$ receives an encryption of m_0^* for S_0^* and in the last game (Game_0^1), $\mathcal{A}$ receives an encryption of m_1^* for S_1^*.

$\mathbf{Game}_0^0$: corresponds to the game given in Definition 3.4 when the challenge bit b^* is fixed to 0. The interaction between $\mathcal{A}$ and $\mathcal{C}$ during *Setup*, *Phase 1*, and *Phase 2* follow exactly as specified in Definition 3.4. During *Challenge*, $\mathcal{A}$ gives $\mathcal{C}$ two equal length messages $m_0^*, m_1^* \in \mathcal{MSP}$ and two equal length sets of user identities $S_0^*, S_1^* \subseteq U$ with the restriction that $\mathsf{Rev} \cap (S_0^* \cup S_1^*) = \emptyset$, where Rev is the set of users that $\mathcal{A}$ corrupted during *Phase 1*. $\mathcal{C}$ computes the challenge ciphertext c^*, which will subsequently be sent to $\mathcal{A}$, as follows:

1. For $j \coloneqq 1$ to l_0, compute $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathsf{HID}_j^0, m_0^*)$.
2. Choose $\widetilde{m} \leftarrow\$ \mathcal{MSP}$.
3. For $j \coloneqq l_0 + 1$ to L, compute $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathtt{dummy}, \widetilde{m})$.
4. Set $c^* \coloneqq (c_{\pi(1)}, \ldots, c_{\pi(L)})$, where $\pi : \{1, \ldots, L\} \to \{1, \ldots, L\}$ is a random permutation.

Eventually, $\mathcal{A}$ outputs a bit b and wins if $b = 0$.

$\mathbf{Game}_h^0$ ($\mathbf{1 \le h \le l_0}$): is similar to Game_{h-1}^0, but $\mathcal{C}$ computes the challenge ciphertext c^* as follows:

1. For $j \coloneqq 1$ to $l_0 - h$, compute $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathsf{HID}_j^0, m_0^*)$.
2. Choose $\widetilde{m} \leftarrow\$ \mathcal{MSP}$.
3. For $j \coloneqq l_0 - h + 1$ to L, compute $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathtt{dummy}, \widetilde{m})$.
4. Set $c^* \coloneqq (c_{\pi(1)}, \ldots, c_{\pi(L)})$, where $\pi : \{1, \ldots, L\} \to \{1, \ldots, L\}$ is a random permutation.

At the end, $\mathcal{A}$ outputs a bit b and wins if $b = 0$.

$\mathbf{Game}_{l_1}^1$: is identical to $\mathrm{Game}_{l_0}^0$

$\mathbf{Game}_k^1$ ($\mathbf{0 \le k < l_1}$): is similar to Game_{k+1}^1, but the challenge ciphertext c^* is now computed by $\mathcal{C}$ as follows:

1. For $j \coloneqq 1$ to $l_1 - k$, compute $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathsf{HID}_j^1, m_1^*)$.
2. Choose $\widetilde{m} \leftarrow\$ \mathcal{MSP}$.
3. For $j \coloneqq l_1 - k + 1$ to L, compute $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathtt{dummy}, \widetilde{m})$.
4. Set $c^* \coloneqq (c_{\pi(1)}, \ldots, c_{\pi(L)})$, where $\pi : \{1, \ldots, L\} \to \{1, \ldots, L\}$ is a random permutation.

Finally, $\mathcal{A}$ outputs a bit b and wins if $b = 0$.

For $0 \le i \le l_0$ and $0 \le j \le l_1$, let $\mathsf{Adv}_{\mathcal{A},\Pi}^{0,i}$ and $\mathsf{Adv}_{\mathcal{A},\Pi}^{1,j}$ denote $\mathcal{A}$'s advantage of winning Game_i^0 and Game_j^1 respectively. In Lemma A.1, we show that if the underlying AIBE scheme is (t, q_{sk}, ϵ)-AIBE-CPA secure, then $\mathcal{A}$'s advantage of distinguishing Game_{h-1}^0 from Game_h^0 is at most ϵ. Similarly, Lemma A.2 states that under similar conditions $\mathcal{A}$'s advantage of distinguishing Game_{k+1}^1 from Game_k^1 is at most ϵ. Therefore,

$$\begin{aligned}\left|\mathsf{Adv}_{\mathcal{A},\Pi}^{0,0} - \mathsf{Adv}_{\mathcal{A},\Pi}^{1,0}\right| &\le \epsilon\,(l_0 + l_1) \\ &\le 2\,\epsilon\,L \\ &\le 2\,\epsilon\,r \log\left(\frac{N}{r}\right).\end{aligned}$$

■

Lemma A.1: *For $1 \leq h \leq l_0$, if the underlying AIBE scheme Π' is (t, q_{sk}, ϵ)-AIBE-CPA-secure, then $\mathcal{A}$'s advantage of distinguishing $Game^0_{h-1}$ from $Game^0_h$ is at most ϵ. In other words,*

$$\left|\mathsf{Adv}^{0,h-1}_{\mathcal{A},\Pi} - \mathsf{Adv}^{0,h}_{\mathcal{A},\Pi}\right| \leq \epsilon. \qquad \square$$

Proof. We build a PPT adversary $\mathcal{B}$ that runs the AIBE-IND-CPA game with its challenger $\mathcal{C}'$ as follows. First, $\mathcal{B}$ receives the master public key MPK' of the AIBE scheme from $\mathcal{C}'$. Next, $\mathcal{B}$ internally executes the oABE-IND-CPA game with $\mathcal{A}$ in order to gain advantage in the AIBE-IND-CPA game. The specifics of the interaction between $\mathcal{C}'$, $\mathcal{B}$, and $\mathcal{A}$ are given below.

Setup: $\mathcal{B}$ forwards MPK' to $\mathcal{A}$. $\mathcal{B}$ also initializes the set of revoked users Rev to be empty.

Phase 1: When $\mathcal{A}$ invokes a secret-key query for user i, first, $\mathcal{B}$ computes HID_i, which is the hierarchical identifier associated with the user i in the binary tree $\mathcal{T}$. Next, for $k \coloneqq 1$ to $n+1$, $\mathcal{B}$ obtains the secret key $sk_{i,k}$ of the identity $\mathsf{HID}_{i|k}$ from its challenger $\mathcal{C}'$. After adding i to Rev, $\mathcal{B}$ sends to $\mathcal{A}$ the secret key of the user i as $sk_i \coloneqq (sk_{i,1}, \ldots, sk_{i,n+1})$.

Challenge: $\mathcal{B}$ receives from $\mathcal{A}$ two equal length messages $m^*_0, m^*_1 \in \mathcal{MSP}$ and two equal length sets of user identities $S^*_0, S^*_1 \subseteq U$ with the restriction that $\mathsf{Rev} \cap (S^*_0 \cup S^*_1) = \emptyset$. $\mathcal{B}$ draws $\widetilde{m} \leftarrow\!\!\$\ \mathcal{MSP}$ and computes the components of its challenge query as follows:

$$id'_0 = \mathsf{HID}^0_{l_0-h+1}, \quad id'_1 = \mathtt{dummy} \qquad m'_0 = m^*_0, \quad m'_1 = \widetilde{m}$$

Observe that the condition $\mathsf{Rev} \cap (S^*_0 \cup S^*_1) = \emptyset$, together with the key assignment strategy of the CS method guarantees that the identity id'_0 hadn't been queried to $\mathcal{B}$'s extraction oracle, and thus this is a valid challenge query to $\mathcal{C}'$.

$\mathcal{B}$ sends the two identities id'_0, id'_1 and the two messages m'_0, m'_1 as the challenge query to $\mathcal{C}'$. $\mathcal{C}'$ picks a random bit $b' \in \{0,1\}$ and sends $c' \leftarrow \mathsf{Enc}(\mathsf{MPK}', id'_{b'}, m'_{b'})$ to $\mathcal{B}$.

Finally, $\mathcal{B}$ computes the challenge ciphertext c^*, which is eventually sent to $\mathcal{A}$, as follows:

1. For $j \coloneqq 1$ to $l_0 - h$, compute $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathsf{HID}^0_j, m^*_0)$.
2. Set $c_{l_0-h+1} \coloneqq c'$.
3. For $j \coloneqq l_0 - h + 2$ to L, compute $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathtt{dummy}, \widetilde{m})$.
4. Set $c^* \coloneqq (c_{\pi(1)}, \ldots, c_{\pi(L)})$, where $\pi : \{1, \ldots, L\} \to \{1, \ldots, L\}$ is a random permutation.

Phase 2: This phase is handled similarly to *Phase 1* with the usual restriction that $\mathcal{A}$ does *not* invoke a secret-key query i such that $i \in S^*_0 \cup S^*_1$.

Guess: $\mathcal{A}$ outputs a guess b and $\mathcal{B}$ passes this bit as its guess for b' to $\mathcal{C}'$.

Observe that, by construction, it holds that if $\mathcal{C}'$ chooses $b' = 0$, then $\mathcal{B}$ is playing Game^0_{h-1}, whereas if $b' = 1$, then $\mathcal{B}$ is playing Game^0_h. Therefore, $\mathcal{B}$'s AIBE-IND-CPA advantage is equivalent to $\mathcal{A}$'s advantage in distinguishing Game^0_{h-1} from Game^0_h. More formally,

$$\left|\mathsf{Adv}^{0,h-1}_{\mathcal{A},\Pi} - \mathsf{Adv}^{0,h}_{\mathcal{A},\Pi}\right| = \mathsf{Adv}^{\text{AIBE-IND-CPA}}_{\mathcal{B},\Pi'} \leq \epsilon. \qquad \blacksquare$$

Lemma A.2: *For $0 \leq k < l_1$, if the underlying AIBE scheme Π' is (t, q_{sk}, ϵ)-AIBE-CPA-secure, then $\mathcal{A}$'s advantage of distinguishing Game^1_{k+1} from Game^1_k is at most ϵ. More precisely,*

$$\left|\mathsf{Adv}^{1,k+1}_{\mathcal{A},\Pi} - \mathsf{Adv}^{1,k}_{\mathcal{A},\Pi}\right| \leq \epsilon. \qquad \square$$

Proof. The argument is analogous to the proof of Lemma A.1, and is therefore omitted. ■

A.2 Proof of Theorem 4.2

Proof. We organize our proof as a sequence of games, $\text{Game}^0_0, \ldots, \text{Game}^0_{l_0} \equiv \text{Game}^1_{l_1}, \ldots, \text{Game}^1_0$, between the adversary $\mathcal{A}$ and the challenger $\mathcal{C}$. In the first game (Game^0_0), $\mathcal{A}$ receives an encryption of m^*_0 for S^*_0 and in the last game (Game^1_0), $\mathcal{A}$ receives an encryption of m^*_1 for S^*_1.

Game^0_0: corresponds to the game given in Definition 3.3 when the challenge bit b^* is fixed to 0. The interaction between $\mathcal{A}$ and $\mathcal{C}$ during *Setup*, *Phase 1*, and *Phase 2* follow exactly as specified in Definition 3.3. During *Challenge* phase, $\mathcal{A}$ gives $\mathcal{C}$ two equal length messages $m^*_0, m^*_1 \in \mathcal{MSP}$ and two equal length sets of user identities $S^*_0, S^*_1 \subseteq U$ with the restriction that $\mathsf{Rev} \cap (S^*_0 \cup S^*_1) = \emptyset$, where Rev is the set of users that $\mathcal{A}$ corrupted during *Phase 1*. $\mathcal{C}$ computes the challenge ciphertext c^*, which will subsequently be sent to $\mathcal{A}$, as follows:

1. Generate $(\mathsf{VK}, \mathsf{SK}) \leftarrow \mathsf{Gen}(1^\lambda)$.
2. For $j \coloneqq 1$ to l_0, compute $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathsf{HID}^0_j, \mathsf{VK} \| m^*_0)$.
3. Choose a random string $\widetilde{m}$ of the same length as $\mathsf{VK} \| m^*_0$.
4. For $j \coloneqq l_0 + 1$ to L, compute $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathtt{dummy}, \widetilde{m})$.
5. Set $\widehat{c} \coloneqq (c_{\pi(1)}, \ldots, c_{\pi(L)})$, where $\pi : \{1, \ldots, L\} \to \{1, \ldots, L\}$ is a random permutation.
6. Generate $\sigma \leftarrow \mathsf{Sign}(\mathsf{SK}, \mathsf{VK} \| \widehat{c})$, and set $c^* \coloneqq (\sigma, \mathsf{VK}, \widehat{c})$.

Eventually, $\mathcal{A}$ outputs a bit b and wins if $b = 0$.

Game^0_h ($1 \leq h \leq l_0$): is similar to Game^0_{h-1}, but $\mathcal{C}$ computes the challenge ciphertext c^* as follows:

1. Generate $(\mathsf{VK}, \mathsf{SK}) \leftarrow \mathsf{Gen}(1^\lambda)$.
2. For $j \coloneqq 1$ to $l_0 - h$, compute $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathsf{HID}^0_j, \mathsf{VK} \| m^*_0)$.
3. Choose a random string $\widetilde{m}$ of the same length a $\mathsf{VK} \| m^*_0$.
4. For $j \coloneqq l_0 - h + 1$ to L, compute $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathtt{dummy}, \widetilde{m})$.
5. Set $\widehat{c} \coloneqq (c_{\pi(1)}, \ldots, c_{\pi(L)})$, where $\pi : \{1, \ldots, L\} \to \{1, \ldots, L\}$ is a random permutation.
6. Generate $\sigma \leftarrow \mathsf{Sign}(\mathsf{SK}, \mathsf{VK} \| \widehat{c})$, and set $c^* \coloneqq (\sigma, \mathsf{VK}, \widehat{c})$.

At the end, $\mathcal{A}$ outputs a bit b and wins if $b = 0$.

$\text{Game}^1_{l_1}$: is identical to $\text{Game}^0_{l_0}$

Game^1_k ($0 \leq k < l_1$): is similar to Game^1_{k+1}, but c^* is now computed by $\mathcal{C}$ as follows:

1. Generate $(\mathsf{VK}, \mathsf{SK}) \leftarrow \mathsf{Gen}(1^\lambda)$.

2. For $j \coloneqq 1$ to $l_1 - k$, compute $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathsf{HID}_j^1, \mathsf{VK} \| m_1^*)$.
3. Choose a random string $\widetilde{m}$ of the same length as $\mathsf{VK} \| m_1^*$.
4. For $j \coloneqq l_1 - k + 1$ to L, compute $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathtt{dummy}, \widetilde{m})$.
5. Set $\widehat{c} \coloneqq (c_{\pi(1)}, \ldots, c_{\pi(L)})$, where $\pi : \{1, \ldots, L\} \to \{1, \ldots, L\}$ is a random permutation.
6. Generate $\sigma \leftarrow \mathsf{Sign}(\mathsf{SK}, \mathsf{VK} \| \widehat{c})$, and set $c^* \coloneqq (\sigma, \mathsf{VK}, \widehat{c})$.

Finally, $\mathcal{A}$ outputs a bit b and wins if $b = 0$.

For $0 \le i \le l_0$ and $0 \le j \le l_1$, let $\mathsf{Adv}_{\mathcal{A},\Pi}^{0,i}$ and $\mathsf{Adv}_{\mathcal{A},\Pi}^{1,j}$ denote $\mathcal{A}$'s advantage of winning Game_i^0 and Game_j^1 respectively. In Lemma A.3, we show that if the underlying one-time signature scheme and AIBE scheme are respectively (t, ϵ_1)-strongly unforgeable and $(t, q_{sk}, q_d, \epsilon_2)$-AIBE-CCA-secure, then $\mathcal{A}$'s advantage of distinguishing Game_{h-1}^0 from Game_h^0 is at most $\epsilon_1 + \epsilon_2$. Similarly, Lemma A.4 states that under analogous conditions $\mathcal{A}$'s advantage of distinguishing Game_{k+1}^1 from Game_k^1 is again at most $\epsilon_1 + \epsilon_2$. Therefore,

$$\begin{aligned}\left|\mathsf{Adv}_{\mathcal{A},\Pi}^{0,0} - \mathsf{Adv}_{\mathcal{A},\Pi}^{1,0}\right| &\le (\epsilon_1 + \epsilon_2)\,(l_0 + l_1) \\ &\le 2\,(\epsilon_1 + \epsilon_2)\,L \\ &\le 2\,(\epsilon_1 + \epsilon_2)\,r \log\!\left(\frac{N}{r}\right).\end{aligned}$$

■

Lemma A.3: *For $1 \le h \le l_0$, if the underlying one-time signature scheme Σ is (t, ϵ_1)-strongly unforgeable and the AIBE scheme Π' is $(t, q_{sk}, q_d, \epsilon_2)$-AIBE-CCA-secure, then $\mathcal{A}$'s advantage of distinguishing Game_{h-1}^0 from Game_h^0 is at most $\epsilon_1 + \epsilon_2$:*

$$\left|\mathsf{Adv}_{\mathcal{A},\Pi}^{0,h-1} - \mathsf{Adv}_{\mathcal{A},\Pi}^{0,h}\right| \le (\epsilon_1 + \epsilon_2).$$

□

Proof. We build a PPT adversary $\mathcal{B}$ that runs the AIBE-IND-CCA game with its challenger $\mathcal{C}'$ as follows. First, $\mathcal{B}$ receives the master public key MPK' of the AIBE scheme from $\mathcal{C}'$. Next, $\mathcal{B}$ internally executes the oABE-IND-CCA game with $\mathcal{A}$ in order to gain advantage in the AIBE-IND-CCA game. The specifics of the interaction between $\mathcal{C}'$, $\mathcal{B}$, and $\mathcal{A}$ are given below.

Setup: $\mathcal{B}$ forwards MPK' to $\mathcal{A}$. $\mathcal{B}$ also initializes the set of revoked users Rev to be empty.

Phase 1: When $\mathcal{A}$ invokes a secret-key query for user i, first, $\mathcal{B}$ computes HID_i, which is the hierarchical identifier associated with the user i in the binary tree $\mathcal{T}$. Next, for $k \coloneqq 1$ to $n+1$, $\mathcal{B}$ obtains the secret key $sk_{i,k}$ of the identity $\mathsf{HID}_{i|k}$ from its challenger $\mathcal{C}'$. After adding i to Rev, $\mathcal{B}$ sends to $\mathcal{A}$ the secret key of the user i as $sk_i \coloneqq (sk_{i,1}, \ldots, sk_{i,n+1})$.

When $\mathcal{A}$ invokes a decryption query $(i, c = (\sigma, \mathsf{VK}, \widehat{c} = (c_1, \ldots, c_L)))$, $\mathcal{B}$ computes HID_i, and for each $k \coloneqq 1$ to $n+1$, $\mathcal{B}$ proceeds as follows:

- If $\mathcal{B}$ obtained the secret key $sk_{i,k}$ corresponding to the identity $\mathsf{HID}_{i|k}$ in the process of responding to a previous secret-key query, then $\mathcal{B}$ attempts to decrypt in turn all ciphertext components $c_1, \ldots, c_L$ in $\widehat{c}$ using the secret key $sk_{i,k}$. If any of these decryption attempts yield a non-$\perp$ value $\mathsf{VK} \| m$, then $\mathcal{B}$ returns m to $\mathcal{A}$ if $\mathsf{Vrfy}(\mathsf{VK}, \sigma, \mathsf{VK} \| \widehat{c})$. Otherwise, $\mathcal{B}$ continues to next k.

- If $\mathcal{B}$ did not obtain the secret key $sk_{i,k}$ of the identity $\mathsf{HID}_{i|k}$ from an earlier secret-key query, then $\mathcal{B}$ makes L decryption queries to its challenger $\mathcal{C}'$, one for each ciphertext component $c_1, \ldots, c_L$, all under identity $\mathsf{HID}_{i|k}$. If any of these decryption queries return a non-$\perp$ value $\mathsf{VK}\|m$, then $\mathcal{B}$ returns m to $\mathcal{A}$ if $\mathsf{Vrfy}(\mathsf{VK}, \sigma, \mathsf{VK}\|\widehat{c})$. Otherwise, $\mathcal{B}$ continues to next j.

If all the above decryption attempts return $\perp$, then $\mathcal{B}$ returns $\perp$ to $\mathcal{A}$.

Challenge: $\mathcal{B}$ receives from $\mathcal{A}$ two equal length messages $m_0^*, m_1^* \in \mathcal{MSP}$ and two equal length sets of user identities $S_0^*, S_1^* \subseteq U$ with the restriction that $\mathsf{Rev} \cap (S_0^* \cup S_1^*) = \emptyset$. $\mathcal{B}$ generates $(\mathsf{VK}, \mathsf{SK}) \leftarrow \mathsf{Gen}(1^\lambda)$, selects a random string $\widetilde{m}$ of the same length as $\mathsf{VK}\|m_0^*$, and sets:

$$id_0' = \mathsf{HID}_{l_0-h+1}^0, \quad id_1' = \mathtt{dummy} \qquad m_0' = \mathsf{VK}\|m_0^*, \quad m_1' = \widetilde{m}$$

Next, $\mathcal{B}$ sends the two identities id_0', id_1' and the two messages m_0', m_1' as the challenge query to $\mathcal{C}'$. $\mathcal{C}'$ picks a random bit $b' \in \{0,1\}$ and responds to $\mathcal{B}$ with $c' \leftarrow \mathsf{Enc}(\mathsf{MPK}', id_{b'}', m_{b'}')$.

Finally, $\mathcal{B}$ computes the challenge ciphertext c^*, which is eventually sent to $\mathcal{A}$, as follows,

1. For $j \coloneqq 1$ to $l_0 - h$, compute $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathsf{HID}_j^0, \mathsf{VK}\|m_0^*)$.
2. Set $c_{l_0-h+1} \coloneqq c'$.
3. For $j \coloneqq l_0 - h + 2$ to L, compute $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathtt{dummy}, \widetilde{m})$.
4. Set $\widehat{c} \coloneqq (c_{\pi(1)}, \ldots, c_{\pi(L)})$, where $\pi : \{1, \ldots, L\} \to \{1, \ldots, L\}$ is a random permutation.
5. Generate $\sigma \leftarrow \mathsf{Sign}(\mathsf{SK}, \mathsf{VK}\|\widehat{c})$, and set $c^* \coloneqq (\sigma, \mathsf{VK}, \widehat{c})$.

Phase 2: Secret-key queries are handled similarly to *Phase 1*, with the usual restriction that $\mathcal{A}$ does *not* invoke a secret-key query i such that $i \in S_0^* \cup S_1^*$.

As for decryption queries, $\mathcal{B}$ replies to $(i, c = (\sigma, \mathsf{VK}, \widehat{c} = (c_1, \ldots, c_L)))$, according to one of the following cases:

- If $c = c^*$ and $i \notin S_0^* \cup S_1^*$, then $\mathcal{B}$ proceeds as in *Phase 1*. (Note that in this case $\mathcal{B}$'s output will be $\perp$, as it should be.)
- If $c = c^*$, and $i \in S_0^* \cup S_1^*$, $\mathcal{B}$ just rejects, since $\mathcal{A}$ is submitting an invalid query.
- If $c \neq c^*$ and $i \notin S_0^*$, then $\mathcal{B}$ proceeds as in *Phase 1*.
- If $c \neq c^*$ and $i \in S_0^*$, then $\mathcal{B}$ computes HID_i, and proceeds as follows:
  - $\diamond$ If for all $k \coloneqq 1$ to $n+1$, it is the case that $\mathsf{HID}_{i|k} \neq \mathsf{HID}_{l_0-h+1}^0$, then $\mathcal{B}$ proceeds as in *Phase 1* (Case b.). Observe that the condition $\forall k \in [1, n+1] : \mathsf{HID}_{i|k} \neq \mathsf{HID}_{l_0-h+1}^0$ ensures that all the decryption queries that $\mathcal{B}$ will make to its challenger $\mathcal{C}'$ in the process of responding to $\mathcal{A}$'s queries are allowed.
  - $\diamond$ If $\exists\ k \in [1, n+1]$ such that $\mathsf{HID}_{i|k} = \mathsf{HID}_{l_0-h+1}^0$, and c' does not appear among the ciphertext components of $\widehat{c}$, then again $\mathcal{B}$ proceeds as in *Phase 1* (Case b.). Observe that the condition that $\widehat{c}$ does not contain c' ensures that also in this case all the decryption queries that $\mathcal{B}$ will make to its challenger $\mathcal{C}'$ in the process of responding to $\mathcal{A}$'s queries are allowed.

- ◇ If $\exists\ k \in [1, n+1]$ such that $\mathsf{HID}_{i|k} = \mathsf{HID}^0_{l_0-h+1}$, but c' appears among the ciphertext components of $\widehat{c}$, then $\mathcal{B}$ outputs $\perp$. To see that $\perp$ is the correct reply, observe that in the real oABE-IND-CCA game, a decryption query (i, c) of this type will trigger decryption of the c' component. Since by construction c' is the encryption of $\mathsf{VK}\|m^*_0$, and $c \neq c^*$, by the unforgeability of the underlying one-time signature scheme, the verification test of Step 1aii of the decryption algorithm would fail, thus yielding $\perp$ as output.

Guess: $\mathcal{A}$ outputs a guess b and $\mathcal{B}$ passes this bit as its guess for b' to $\mathcal{C}'$.

Observe that, by construction, it holds that if $\mathcal{C}'$ chooses $b' = 0$, then $\mathcal{B}$ is playing Game^0_{h-1}, whereas if $b' = 1$, then $\mathcal{B}$ is playing Game^0_h. Therefore, up to forgeries of the underlying one-time signature scheme, $\mathcal{B}$'s AIBE-IND-CCA advantage is essentially $\mathcal{A}$'s advantage in distinguishing Game^0_{h-1} from Game^0_h:

$$\left|\mathsf{Adv}^{0,h-1}_{\mathcal{A},\Pi} - \mathsf{Adv}^{0,h}_{\mathcal{A},\Pi}\right| \leq (\epsilon_1 + \epsilon_2).$$

■

Lemma A.4: *For $0 \leq k < l_1$, if the underlying one-time signature scheme Σ is (t, ϵ_1)-strongly unforgeable and the underlying AIBE scheme Π' is $(t, q_{sk}, q_d, \epsilon_2)$-AIBE-CCA-secure, then $\mathcal{A}$'s advantage of distinguishing Game^1_{k+1} from Game^1_k is at most ϵ. More precisely,*

$$\left|\mathsf{Adv}^{1,k+1}_{\mathcal{A},\Pi} - \mathsf{Adv}^{1,k}_{\mathcal{A},\Pi}\right| \leq (\epsilon_1 + \epsilon_2).$$

□

Proof. The argument is analogous to the proof of Lemma A.3, and is therefore omitted. ■

A.3 Proof of Theorem 4.3

Proof. We organize our proof as a sequence of games between the adversary $\mathcal{A}$ and the challenger $\mathcal{C}$ as follows:

$$\mathrm{Game}^0_0, \overline{\mathrm{Game}}^0_1, \mathrm{Game}^0_1, \ldots, \overline{\mathrm{Game}}^0_{l_0}, \mathrm{Game}^0_{l_0} \equiv \mathrm{Game}^1_{l_1}, \overline{\mathrm{Game}}^1_{l_1}, \ldots, \mathrm{Game}^1_1, \overline{\mathrm{Game}}^1_1, \mathrm{Game}^1_0.$$

In the first game (Game^0_0), $\mathcal{A}$ receives an encryption of m^*_0 for S^*_0 and in the last game (Game^1_0), $\mathcal{A}$ receives an encryption of m^*_1 for S^*_1.

Game^0_0: corresponds to the game given in Definition 3.3 when the challenge bit b^* is fixed to 0. The interaction between $\mathcal{A}$ and $\mathcal{C}$ during *Setup*, *Phase 1*, and *Phase 2* follow exactly as specified in Definition 3.3. During *Challenge*, $\mathcal{A}$ gives $\mathcal{C}$ two equal length messages $m^*_0, m^*_1 \in \mathcal{MSP}$ and two equal length sets of user identities $S^*_0, S^*_1 \subseteq U$ with the restriction that $\mathsf{Rev} \cap (S^*_0 \cup S^*_1) = \emptyset$, where Rev is the set of users that $\mathcal{A}$ corrupted during *Phase 1*. $\mathcal{C}$ computes the challenge ciphertext c^*, which will subsequently be sent to $\mathcal{A}$, as follows:

1. Generate $(\mathsf{VK}, \mathsf{SK}) \leftarrow \mathsf{Gen}(1^\lambda)$.
2. Draw $s \leftarrow\$\ \mathbb{Z}_q$, and compute $\overline{c}_0 \coloneqq g^s$.
3. For $j \coloneqq 1$ to l_0, compute $\overline{c}_j \coloneqq H(A^s_{\mathsf{HID}^0_j})$, $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathsf{HID}^0_j, \mathsf{VK}\|A^s_{\mathsf{HID}^0_j}\|m^*_0)$.
4. Choose a random string $\widetilde{m}$ of the same length as $\mathsf{VK}\|\overline{c}_0\|m^*_0$.

5. For $j \coloneqq l_0+1$ to L, set $s_j \leftarrow\$ \mathbb{Z}_q$, and compute $\overline{c}_j \coloneqq H(g^{s_j})$, $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathtt{dummy}, \widetilde{m})$.
6. Set $\widehat{c} \coloneqq (\overline{c}_0, (\overline{c}_{\pi(1)}, c_{\pi(1)}), \ldots, (\overline{c}_{\pi(L)}, c_{\pi(L)}))$, where $\pi : \{1,\ldots,L\} \to \{1,\ldots,L\}$ is a random permutation.
7. Generate $\sigma \leftarrow \mathsf{Sign}(\mathsf{SK}, \mathsf{VK}\|\widehat{c})$, and set $c^* \coloneqq (\sigma, \mathsf{VK}, \widehat{c})$.

Eventually, $\mathcal{A}$ outputs a bit b and wins if $b = 0$.

$\overline{\text{Game}}^0_h$($1 \le h \le l_0$): is similar to Game^0_{h-1}, but $\mathcal{C}$ computes the challenge ciphertext c^* as follows:

1. Generate $(\mathsf{VK}, \mathsf{SK}) \leftarrow \mathsf{Gen}(1^\lambda)$.
2. Draw $s \leftarrow\$ \mathbb{Z}_q$, and compute $\overline{c}_0 \coloneqq g^s$.
3. For $j \coloneqq 1$ to $l_0 - h$, compute $\overline{c}_j \coloneqq H(A^s_{\mathsf{HID}^0_j})$, $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathsf{HID}^0_j, \mathsf{VK}\|A^s_{\mathsf{HID}^0_j}\|m^*_0)$.
4. Choose a random string $\widetilde{m}$ of the same length as $\mathsf{VK}\|\overline{c}_0\|m^*_0$.
5. Compute $\overline{c}_{l_0-h+1} \coloneqq H(A^s_{\mathsf{HID}^0_{l_0-h+1}})$, $c_{l_0-h+1} \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathtt{dummy}, \widetilde{m})$.
6. For $j \coloneqq l_0-h+2$ to L, set $s_j \leftarrow\$ \mathbb{Z}_q$, and compute $\overline{c}_j \coloneqq H(g^{s_j})$, $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathtt{dummy}, \widetilde{m})$.
7. Set $\widehat{c} \coloneqq (\overline{c}_0, (\overline{c}_{\pi(1)}, c_{\pi(1)}), \ldots, (\overline{c}_{\pi(L)}, c_{\pi(L)}))$, where $\pi : \{1,\ldots,L\} \to \{1,\ldots,L\}$ is a random permutation.
8. Generate $\sigma \leftarrow \mathsf{Sign}(\mathsf{SK}, \mathsf{VK}\|\widehat{c})$, and set $c^* \coloneqq (\sigma, \mathsf{VK}, \widehat{c})$.

At the end, $\mathcal{A}$ outputs a bit b and wins if $b = 0$.

Game^0_h($1 \le h \le l_0$): is similar to $\overline{\text{Game}}^0_h$, but $\mathcal{C}$ computes the challenge ciphertext c^* as follows:

1. Generate $(\mathsf{VK}, \mathsf{SK}) \leftarrow \mathsf{Gen}(1^\lambda)$.
2. Draw $s \leftarrow\$ \mathbb{Z}_q$, and compute $\overline{c}_0 \coloneqq g^s$.
3. For $j \coloneqq 1$ to $l_0 - h$, compute $\overline{c}_j \coloneqq H(A^s_{\mathsf{HID}^0_j})$, $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathsf{HID}^0_j, \mathsf{VK}\|A^s_{\mathsf{HID}^0_j}\|m^*_0)$.
4. Choose a random string $\widetilde{m}$ of the same length as $\mathsf{VK}\|\overline{c}_0\|m^*_0$.
5. For $j \coloneqq l_0-h+1$ to L, set $s_j \leftarrow\$ \mathbb{Z}_q$, and compute $\overline{c}_j \coloneqq H(g^{s_j})$, $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathtt{dummy}, \widetilde{m})$.
6. Set $\widehat{c} \coloneqq (\overline{c}_0, (\overline{c}_{\pi(1)}, c_{\pi(1)}), \ldots, (\overline{c}_{\pi(L)}, c_{\pi(L)}))$, where $\pi : \{1,\ldots,L\} \to \{1,\ldots,L\}$ is a random permutation.
7. Generate $\sigma \leftarrow \mathsf{Sign}(\mathsf{SK}, \mathsf{VK}\|\widehat{c})$, and set $c^* \coloneqq (\sigma, \mathsf{VK}, \widehat{c})$.

Finally, $\mathcal{A}$ outputs a bit b and wins if $b = 0$.

Game$^1_{l_1}$: is identical to $\text{Game}^0_{l_0}$

$\overline{\text{Game}}^1_k$($1 \le k \le l_1$): is similar to Game^1_k, with the challenge ciphertext c^* computed by $\mathcal{C}$ as follows:

1. Generate $(\mathsf{VK}, \mathsf{SK}) \leftarrow \mathsf{Gen}(1^\lambda)$.
2. Draw $s \leftarrow\$ \mathbb{Z}_q$, and compute $\overline{c}_0 \coloneqq g^s$.
3. For $j \coloneqq 1$ to $l_1 - k$, compute $\overline{c}_j \coloneqq H(A^s_{\mathsf{HID}^1_j})$, $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathsf{HID}^1_j, \mathsf{VK}\|A^s_{\mathsf{HID}^1_j}\|m^*_1)$.

4. Choose a random string $\widetilde{m}$ of the same length as $\mathsf{VK}\|\overline{c}_0\|m_1^*$.
5. Compute $\overline{c}_{l_1-k+1} \coloneqq H(A^s_{\mathsf{HID}^1_{l_1-k+1}})$, $c_{l_1-k+1} \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathtt{dummy}, \widetilde{m})$.
6. For $j \coloneqq l_1-k+2$ to L, set $s_j \leftarrow\$ \mathbb{Z}_q$, and compute $\overline{c}_j \coloneqq H(g^{s_j})$, $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathtt{dummy}, \widetilde{m})$.
7. Set $\widehat{c} \coloneqq (\overline{c}_0, (\overline{c}_{\pi(1)}, c_{\pi(1)}), \ldots, (\overline{c}_{\pi(L)}, c_{\pi(L)}))$, where $\pi : \{1,\ldots,L\} \to \{1,\ldots,L\}$ is a random permutation.
8. Generate $\sigma \leftarrow \mathsf{Sign}(\mathsf{SK}, \mathsf{VK}\|\widehat{c})$, and set $c^* \coloneqq (\sigma, \mathsf{VK}, \widehat{c})$.

At last, $\mathcal{A}$ outputs a bit b and wins if $b = 0$.

$\mathbf{Game}^{\mathbf{1}}_{\boldsymbol{k}}$($\mathbf{0 \le k < l_1}$): is similar to $\overline{\mathrm{Game}}^1_{k+1}$, but $\mathcal{C}$ computes the challenge ciphertext c^* as follows:

1. Generate $(\mathsf{VK}, \mathsf{SK}) \leftarrow \mathsf{Gen}(1^\lambda)$.
2. Draw $s \leftarrow\$ \mathbb{Z}_q$, and compute $\overline{c}_0 \coloneqq g^s$.
3. For $j \coloneqq 1$ to $l_1 - k$, compute $\overline{c}_j \coloneqq H(A^s_{\mathsf{HID}^1_j})$, $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathsf{HID}^1_j, \mathsf{VK}\|A^s_{\mathsf{HID}^1_j}\|m_1^*)$.
4. Choose a random string $\widetilde{m}$ of the same length as $\mathsf{VK}\|\overline{c}_0\|m_1^*$.
5. For $j \coloneqq l_1-k+1$ to L, set $s_j \leftarrow\$ \mathbb{Z}_q$, and compute $\overline{c}_j \coloneqq H(g^{s_j})$, $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathtt{dummy}, \widetilde{m})$.
6. Set $\widehat{c} \coloneqq (\overline{c}_0, (\overline{c}_{\pi(1)}, c_{\pi(1)}), \ldots, (\overline{c}_{\pi(L)}, c_{\pi(L)}))$, where $\pi : \{1,\ldots,L\} \to \{1,\ldots,L\}$ is a random permutation.
7. Generate $\sigma \leftarrow \mathsf{Sign}(\mathsf{SK}, \mathsf{VK}\|\widehat{c})$, and set $c^* \coloneqq (\sigma, \mathsf{VK}, \widehat{c})$.

Finally, $\mathcal{A}$ outputs a bit b and wins if $b = 0$.

For $0 \le i_1 \le l_0$, $1 \le i_2 \le l_0$, $0 \le j_1 \le l_1$ and $1 \le j_2 \le l_1$, let $\mathsf{Adv}^{0,i_1}_{\mathcal{A},\Pi}$, $\overline{\mathsf{Adv}}^{0,i_2}_{\mathcal{A},\Pi}$, $\mathsf{Adv}^{1,j_1}_{\mathcal{A},\Pi}$ and $\overline{\mathsf{Adv}}^{1,j_2}_{\mathcal{A},\Pi}$ denote $\mathcal{A}$'s advantage of winning $\mathrm{Game}^0_{i_1}$, $\overline{\mathrm{Game}}^0_{i_2}$, $\mathrm{Game}^1_{j_1}$ and $\overline{\mathrm{Game}}^0_{j_2}$ respectively. In Lemma A.5, we show that if the underlying one-time signature scheme and AIBE scheme are respectively (t, ϵ_1)-strongly unforgeable and $(t, q_{sk}, q_d, \epsilon_2)$-AIBE-CCA-secure, then $\mathcal{A}$'s advantage of distinguishing Game^0_{h-1} from $\overline{\mathrm{Game}}^0_h$ is at most $\epsilon_1 + \epsilon_2$. And, in Lemma A.6, we show that if CDH is (t, ϵ_3)-hard in $\mathbb{G}$ and DDH is efficiently computable in $\mathbb{G}$, then $\mathcal{A}$ has at most ϵ_3 advantage in distinguishing $\overline{\mathrm{Game}}^0_h$ from Game^0_h. Similarly, Lemma A.7 and Lemma A.8 states that under analogous conditions $\mathcal{A}$'s advantages of distinguishing $\overline{\mathrm{Game}}^1_{k+1}$ from Game^1_k, and Game^1_k from $\overline{\mathrm{Game}}^1_k$ is at most $\epsilon_1 + \epsilon_2$ and ϵ_3 respectively. Therefore,

$$\begin{aligned}\left|\mathsf{Adv}^{0,0}_{\mathcal{A},\Pi} - \mathsf{Adv}^{1,0}_{\mathcal{A},\Pi}\right| &\le (\epsilon_1 + \epsilon_2 + \epsilon_3)\,(l_0 + l_1) \\ &\le 2\,(\epsilon_1 + \epsilon_2 + \epsilon_3)\,L \\ &\le 2\,(\epsilon_1 + \epsilon_2 + \epsilon_3)\,r \log\!\left(\frac{N}{r}\right).\end{aligned}$$

■

Lemma A.5: *For $1 \le h \le l_0$, if the underlying one-time signature scheme Σ is (t, ϵ_1)-strongly unforgeable and the AIBE scheme Π' is $(t, q_{sk}, q_d, \epsilon_2)$-AIBE-CCA-secure, then $\mathcal{A}$'s advantage of distinguishing Game^0_{h-1} from $\overline{\mathrm{Game}}^0_h$ is at most $\epsilon_1 + \epsilon_2$:*

$$\left|\mathsf{Adv}^{0,h-1}_{\mathcal{A},\Pi} - \overline{\mathsf{Adv}}^{0,h}_{\mathcal{A},\Pi}\right| \le (\epsilon_1 + \epsilon_2).$$

□

Proof. The proof is very similar to that of Lemma A.3 and is therefore omitted. The only difference which we should be careful about is the new tag system of the ciphertext components. The challenger can trivially compute these tags as specified in the construction of Sect. 4.3 and attach them to the corresponding ciphertext components during the simulation. ■

Lemma A.6: *For $1 \leq h \leq l_0$, if CDH is (t, ϵ_3)-hard in $\mathbb{G}$ and DDH is efficiently computable in $\mathbb{G}$, then $\mathcal{A}$'s advantage of distinguishing $\overline{Game}_h^0$ from $Game_h^0$ is at most ϵ_3:*

$$\left|\overline{\mathsf{Adv}}_{\mathcal{A},\Pi}^{0,h} - \mathsf{Adv}_{\mathcal{A},\Pi}^{0,h}\right| \leq \epsilon_3. \qquad \square$$

Proof. Let F be the event that $\mathcal{A}$ queries the random oracle H at the point $A^s_{\mathsf{HID}^0_{l_0-h+1}}$. By construction, it is clear that

$$\left|\overline{\mathsf{Adv}}_{\mathcal{A},\Pi}^{0,h} - \mathsf{Adv}_{\mathcal{A},\Pi}^{0,h}\right| \leq \Pr[F].$$

We want to show $\Pr[F] \leq \Pr[\text{CDH}] \leq \epsilon_3$. Assuming $\mathcal{A}$ can distinguish $\overline{\text{Game}}_h^0$ from Game_h^0, we build a PPT CDH adversary $\mathcal{B}$ which uses $\mathcal{A}$ as a subroutine. First, $\mathcal{B}$ gets a CDH instance $(g, X = g^x, Y = g^y)$ as input from the CDH challenger. Then, $\mathcal{B}$ simulates the challenger's behavior in Game_h^0 to $\mathcal{A}$ as follows:

Setup: $\mathcal{B}$ simulates the *Setup* phase as in Definition 3.3 except that it sets $A_{\mathsf{HID}^0_{l_0-h+1}} = Y$.

Phase 1: $\mathcal{B}$ handles the secret-key queries as specified in Definition 3.3.

Given a decryption query (i, c), we distinguish two cases. If the node (which is denoted by u for simplicity) with hierarchical identifier $\mathsf{HID}^0_{l_0-h+1}$ is not among the ancestors of the leaf node corresponding to the user i in the tree $\mathcal{T}$, then $\mathcal{B}$ just runs the $\mathsf{Decrypt}$ algorithm in Sect. 4.3. Otherwise, $\mathcal{B}$ still runs the $\mathsf{Decrypt}$ algorithm as in Sect. 4.3, but with one modification. That is, during Step 1, he skips the computation of the tag corresponding to node u. If this modified computation of the $\mathsf{Decrypt}$ algorithm yielded a valid message m, $\mathcal{B}$ simply returns that m. If not, $\mathcal{B}$ proceeds as follows:

1. Denote by sk_u the AIBE secret key of the node u.
2. Parse c as $(\sigma, \mathsf{VK}, \widehat{c} = (\overline{c}_0, (\overline{c}_1, c_1), \ldots, (\overline{c}_L, c_L)))$.
3. For $j \coloneqq 1$ to L,
   a. Compute $m' \coloneqq \mathsf{Dec}(\mathsf{MPK}', sk_u, c_j)$.
   b. If $m' = \mathsf{VK} \| Z \| m$, $\mathsf{Vrfy}(\mathsf{VK}, \sigma, \mathsf{VK} \| \widehat{c})$, the DDH algorithm accepts $(g, \overline{c}_0, Y, Z)$, and $\overline{c}_j = H(Z)$, return m.
4. If Step 3 did not result in a valid m, return $\perp$ (as the original $\mathsf{Decrypt}$ algorithm would have returned).

Challenge: Given two equal length messages $m_0^*, m_1^* \in \mathcal{MSP}$ and two equal length sets of user identities $S_0^*, S_1^* \subseteq U$ with the restriction that $\mathsf{Rev} \cap (S_0^* \cup S_1^*) = \emptyset$, $\mathcal{B}$ computes the challenge ciphertext c^* as follows:

1. Generate $(\mathsf{VK}, \mathsf{SK}) \leftarrow \mathsf{Gen}(1^\lambda)$.

2. Set $\overline{c}_0 \coloneqq X$.
3. For $j \coloneqq 1$ to $l_0 - h$, compute $\overline{c}_j \coloneqq H(X^{a_{\mathsf{HID}^0_j}})$, $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathsf{HID}^0_j, \mathsf{VK} \| X^{a_{\mathsf{HID}^0_j}} \| m^*_0)$.
4. Choose a random string $\widetilde{m}$ of the same length as $\mathsf{VK} \| \overline{c}_0 \| m^*_0$.
5. Compute $\overline{c}_{l_0-h+1} \leftarrow\$ \{0,1\}^\lambda$, $c_{l_0-h+1} \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathtt{dummy}, \widetilde{m})$.
6. For $j \coloneqq l_0 - h + 2$ to L, set $s_j \leftarrow\$ \mathbb{Z}_q$, and compute $\overline{c}_j \coloneqq H(g^{s_j})$, $c_j \leftarrow \mathsf{Enc}(\mathsf{MPK}', \mathtt{dummy}, \widetilde{m})$.
7. Set $\widehat{c} \coloneqq (\overline{c}_0, (\overline{c}_{\pi(1)}, c_{\pi(1)}), \ldots, (\overline{c}_{\pi(L)}, c_{\pi(L)}))$, where $\pi : \{1, \ldots, L\} \to \{1, \ldots, L\}$ is a random permutation.
8. Generate $\sigma \leftarrow \mathsf{Sign}(\mathsf{SK}, \mathsf{VK} \| \widehat{c})$, and set $c^* \coloneqq (\sigma, \mathsf{VK}, \widehat{c})$.

Phase 2: $\mathcal{B}$ handles *Phase 2* as in *Phase 1* with the same restrictions given in Definition 3.2.[3]

Guess: $\mathcal{A}$ outputs a guess b and $\mathcal{B}$ saves it.

When simulating the random oracle H to $\mathcal{A}$, $\mathcal{B}$ picks $R \leftarrow\$ \{0,1\}^\lambda$ as the result and responds consistently. If $\mathcal{A}$ ever makes a random oracle query Z such that the DDH algorithm accepts (g, X, Y, Z), $\mathcal{B}$ halts the computation and outputs Z as its CDH solution.

By construction, $\mathcal{A}$ can distinguish $\overline{\mathrm{Game}}^0_h$ from Game^0_h only if it queries the random oracle on g^{xy} or sends a decryption oracle query with a ciphertext component containing g^{xy}. In both cases, $\mathcal{B}$ suspends the computation and wins the CDH game. Therefore, $\mathcal{A}$'s advantage in distinguishing $\overline{\mathrm{Game}}^0_h$ from Game^0_h is at most ϵ_3. ■

Lemma A.7: *For $0 \leq k < l_1$, if the underlying one-time signature scheme Σ is (t, ϵ_1)-strongly unforgeable and the AIBE scheme Π' is $(t, q_{sk}, q_d, \epsilon_2)$-AIBE-CCA-secure, then $\mathcal{A}$'s advantage of distinguishing $\overline{\mathit{Game}}^1_{k+1}$ from Game^1_k is at most $\epsilon_1 + \epsilon_2$. More precisely,*

$$\left| \overline{\mathsf{Adv}}^{1,k+1}_{\mathcal{A},\Pi} - \mathsf{Adv}^{1,k}_{\mathcal{A},\Pi} \right| \leq (\epsilon_1 + \epsilon_2). \qquad \square$$

Proof. The argument is analogous to the proof of Lemma A.5, and is therefore omitted. ■

Lemma A.8: *For $1 \leq k \leq l_1$, if CDH is (t, ϵ_3)-hard in $\mathbb{G}$ and DDH is efficiently computable in $\mathbb{G}$, then $\mathcal{A}$'s advantage of distinguishing Game^1_k from $\overline{\mathit{Game}}^1_k$ is at most ϵ_3:*

$$\left| \mathsf{Adv}^{1,k}_{\mathcal{A},\Pi} - \overline{\mathsf{Adv}}^{1,k}_{\mathcal{A},\Pi} \right| \leq \epsilon_3. \qquad \square$$

Proof. The argument is analogous to the proof of Lemma A.6, and is therefore omitted. ■

[3] A slight complication is that, post-challenge, the adversary could try to reuse the $\overline{c}_0, \overline{c}_{\pi(1)}, \ldots, \overline{c}_{\pi(L)}$ components from c^*, but combine them with fresh $\tilde{c}_1, \ldots, \tilde{c}_L$ components for some message $\widehat{m}$ of her choice. If a ciphertext so crafted were submitted to the decryption oracle for user u, then $\mathcal{B}$ would invoke the special decryption process described in *Phase 1*, and would be unable to test whether $\overline{c}_j = H(Z)$. However, in order for all the other checks in Step 3b to go through, m' should be equal to $\widehat{\mathsf{VK}} \| Z \| \widehat{m}$, for a Z such that (g, X, Y, Z) is a DDH tuple. Clearly, at that point $\mathcal{B}$ could simply halt its computation, and output Z as its answer to its CDH challenge.

A.4 Proof of Theorem 4.4

Proof. The proof of this theorem follows almost the same structure as that of Theorem 4.3, with the exception that the tags are now created as described in Sect. 4.4. More specifically, as in the proof of Theorem 4.3, we again consider the following sequence of games between the adversary $\mathcal{A}$ and the challenger $\mathcal{C}$:

$$\mathrm{Game}_0^0, \overline{\mathrm{Game}}_1^0, \mathrm{Game}_1^0, \ldots, \overline{\mathrm{Game}}_{l_0}^0, \mathrm{Game}_{l_0}^0 \equiv \mathrm{Game}_{l_1}^1, \overline{\mathrm{Game}}_{l_1}^1, \ldots, \mathrm{Game}_1^1, \overline{\mathrm{Game}}_1^1, \mathrm{Game}_0^1.$$

In the first game (Game_0^0), $\mathcal{A}$ receives an encryption of m_0^* for S_0^* and in the last game (Game_0^1), $\mathcal{A}$ receives an encryption of m_1^* for S_1^*.

Game_0^0 corresponds to the original game as described in Definition 3.2, when the challenge bit b^* is fixed to 0.

$\overline{\mathrm{Game}}_h^0 (1 \leq h \leq l_0)$ is similar to Game_{h-1}^0, except that at position $j = l_0 - h + 1$, $\mathcal{C}$ pairs the correct tag $\overline{c}_j = ((A_{\mathsf{HID}_j}^{\mathsf{VK}} B_{\mathsf{HID}_j})^s, (C_{\mathsf{HID}_j}^{\mathsf{VK}} D_{\mathsf{HID}_j})^s)$ with an encryption c_j of a random string $\widetilde{m}$ of the same length of $\mathsf{VK} \| m_0^*$.

$\mathrm{Game}_h^0 (1 \leq h \leq l_0)$ is similar to $\overline{\mathrm{Game}}_h^0$, but $\mathcal{C}$ computes the challenge ciphertext components for position $j = l_0 - h + 1$ as follows: to create tag $\overline{c}_{l_0-h+1}$, $\mathcal{C}$ uses a random value $s_j \leftarrow\!\!\$\ \mathbb{Z}_q$.

The description of $\overline{\mathrm{Game}}_k^1 (1 \leq k \leq l_1)$, and $\mathrm{Game}_k^1 (0 \leq k < l_1)$ is as above, where we replace m_0^* with m_1^*.

For $0 \leq i_1 \leq l_0$, $1 \leq i_2 \leq l_0$, $0 \leq j_1 \leq l_1$ and $1 \leq j_2 \leq l_1$, let $\mathsf{Adv}_{\mathcal{A},\Pi}^{0,i_1}$, $\overline{\mathsf{Adv}}_{\mathcal{A},\Pi}^{0,i_2}$, $\mathsf{Adv}_{\mathcal{A},\Pi}^{1,j_1}$ and $\overline{\mathsf{Adv}}_{\mathcal{A},\Pi}^{1,j_2}$ denote $\mathcal{A}$'s advantage of winning $\mathrm{Game}_{i_1}^0$, $\overline{\mathrm{Game}}_{i_2}^0$, $\mathrm{Game}_{j_1}^1$ and $\overline{\mathrm{Game}}_{j_2}^0$ respectively.

The proof that $\mathcal{A}$'s advantage of distinguishing Game_{h-1}^0 from $\overline{\mathrm{Game}}_h^0$ is at most $\epsilon_1 + \epsilon_2$ is essentially identical to that of Lemma A.5.

The proof that $\mathcal{A}$'s advantage of distinguishing $\overline{\mathrm{Game}}_h^0$ from Game_h^0 is at most $2\left(\epsilon_3 + \frac{q_d}{2^\lambda}\right)$ (where ϵ_3 is the advantage of breaking DDH in $\mathbb{G}$) is essentially identical to that of Lemma 1 of [27].

Similarly, $\mathcal{A}$'s advantages of distinguishing $\overline{\mathrm{Game}}_{k+1}^1$ from Game_k^1, and Game_k^1 from $\overline{\mathrm{Game}}_k^1$ are at most $\epsilon_1 + \epsilon_2$ and $2\left(\epsilon_3 + \frac{q_d}{2^\lambda}\right)$, respectively. Therefore,

$$\begin{aligned}
\left|\mathsf{Adv}_{\mathcal{A},\Pi}^{0,0} - \mathsf{Adv}_{\mathcal{A},\Pi}^{1,0}\right| &\leq 2\left(\epsilon_1 + \epsilon_2 + 2\left(\epsilon_3 + \frac{q_d}{2^\lambda}\right)\right)(l_0 + l_1) \\
&\leq 2\left(\epsilon_1 + \epsilon_2 + 2\left(\epsilon_3 + \frac{q_d}{2^\lambda}\right)\right) L \\
&\leq 2\left(\epsilon_1 + \epsilon_2 + 2\left(\epsilon_3 + \frac{q_d}{2^\lambda}\right)\right) r \log\left(\frac{N}{r}\right).
\end{aligned}$$

■