Distributed Broadcast Encryption from Bilinear Groups

Dimitris Kolonelos[1,2], Giulio Malavolta[3], and Hoeteck Wee[4,5]

[1]IMDEA Software Institute
[2]Universidad Politecnica de Madrid
[3]Max Planck Institute for Security and Privacy
[4]NTT Research
[5]École Normale Supérieure - PSL

August 3, 2023

Abstract

In this work, we show that obfuscation is not necessary for DBE, and we present two DBE schemes from standard assumptions in prime-order bilinear groups. Our constructions are conceptually simple, satisfy the strong notion of adaptive security, and are concretely efficient. In fact, their performance, in terms of number of group elements and efficiency of the algorithms, is comparable with that of traditional (non distributed) broadcast encryption schemes from bilinear groups.

Distributed broadcast encryption (DBE) improves on the traditional notion of broadcast encryption by eliminating the key-escrow problem: In a DBE system, users generate their own secret keys non-interactively without the help of a trusted party. Then anyone can broadcast a message for a subset S of the users, in such a way that the resulting ciphertext size is sublinear in (and, ideally, independent of) $|S|$. Unfortunately, the only known constructions of DBE requires heavy cryptographic machinery, such as general-purpose indistinguishability obfuscation, or come without a security proof.

In this work, we place DBE on similar footing as traditional (non-distributed) broadcast encryption: We present two practical DBE schemes from standard assumptions in prime-order bilinear groups. Our constructions are conceptually simple, satisfy the strong notion of adaptive security, and are concretely efficient. Their performance, in terms of number of group elements and efficiency of the algorithms, is comparable with that of traditional broadcast encryption schemes from bilinear groups.

Keywords: Pairing-based Cryptography, Broadcast Encryption, Key-Escrow

1 Introduction

In a broadcast encryption (BE) scheme [14] a broadcaster encrypts a message for some subset S of the users who are listening on a broadcast channel. Any user that belongs to the set S can recover the message using their own secret key. The security requirement stipulates that, even if all users not in S collude, they learn nothing about the broadcasted message.

Broadcast encryption has been an active area of research since their introduction in the 1990s, where a major goal is to obtain schemes with short parameters, notably ciphertext size that is sublinear in the total number of users L, so as to minimize bandwidth consumption. In a celebrated work from 2005, Boneh,

Gentry and Waters (BGW) [4] presented a pairing-based broadcast encryption scheme with constant-size ciphertext (ignoring the contribution from the set S). A series of follow-up works [5, 21, 11] showed how to achieve constant-size ciphertext under the standard k-Lin assumption, improving upon the q-type assumption used in BGW, while additionally strengthening the security guarantees from selective to adaptive security. More recent works [2, 1, 10, 34], improving upon [6, 7], showed how to achieve $\mathrm{poly}(\log L)$ total parameter size under stronger, non-falsifiable assumptions.

All of the aforementioned broadcast encryption schemes suffer from the notorious *key escrow* problem: the schemes require a central authority holding a master secret key, that generates and distributes keys for all users of the system. Moreover, the central authority can decrypt *any ciphertext* ever encrypted using this master secret key. Another security concern is that the central authority needs to remain online with a long-term secret key, which constitutes a recurrent single point of failure.

Distributed Broadcast Encryption. To circumvent the key escrow problem [37, 7] introduced the notion of *distributed* broadcast encryption (DBE), where users choose their own public/secret key pairs, and replacing the central authority with a bulletin board of user public keys. This not only solves the key escrow problem, but also captures many applications such as peer-to-peer networks, on-the-fly data sharing, and group messaging.

Wu, Qin, Zhang and Domingo-Ferrer [37] (henceforth WQZD), present a pairing-based scheme for L users with a transparent set-up where each user's public key comprises $O(L^2)$ group elements, and the ciphertext comprises $O(1)$ group elements (more discussion on this in Section 1.2), however without a security proof. Shortly after, Boneh and Zhandry [7] construct a distributed broadcast encryption scheme with $\mathrm{poly}(\log L)$-sized public keys and ciphertext assuming indistinguishability obfuscation, in a stronger model with a one-time trusted sampling of a common reference string (CRS). We stress that the trusted setup only needs to be done once (e.g., with an MPC), that the same CRS can be reused across different systems, and that there is no need to store any long-term secrets, thereby also circumventing the key escrow problem. We regard this mostly as a feasibility result, given the state of affairs for obfuscation [26, 27, 8, 35, 20, 9]. In this work, we address the following question:

Can we construct *simple* and *efficient* distributed broadcast encryption with small ciphertexts?

In particular, we focus on pairing-based schemes, due to the increasing support (e.g., high-quality implementations with strong assurance and performance, on-going IETF standardization) and deployment of pairing-based cryptography.

1.1 Our Results

We construct simple pairing-based distributed broadcast encryption (DBE) schemes where, for a bound of L users:

- Ciphertexts comprise a constant number of group elements, like in BGW;
- Encryption to a set S only requires retrieving $O(|S|)$ group elements from the bulletin board;
- Users only need to store $O(L)$ group elements for decryption;
- Both encryption and decryption take time linear in $|S|$.

	Assumption	$\lvert\mathsf{pp}\rvert$	$\lvert\mathsf{usk}_j\rvert$	$\lvert\mathsf{upk}_j\rvert$	$\lvert\mathsf{ct}\rvert$	$\lvert\mathsf{BB}\rvert$	#Updates
Folklore	PKE	–	1	1	$\lvert S\rvert$	L	–
Folklore	RBE	$\log L$	$\log L$	1	$\lvert S\rvert$	$L \log L$	$\log L$
Boneh-Zhandry	iO and OWF	L	1	1	1	L	L
WQZD	BDHE	L	L	L^2	1	L^3	L
Scheme 1	BDHE	L	1	L	1	L^2	L
Scheme 1 (Log. Updates)	BDHE	L	1	L	$\log L$	L^2	$\log L$
Scheme 2	k-Lin	L^2	1	L	1	L^2	L
Scheme 2 (Log. Updates)	k-Lin	L^2	1	L	$\log L$	L^2	$\log L$

Table 1: Comparison with existing DBE schemes. The notation in the table ignores constants and factors that depend only on the security parameters. $\lvert\mathsf{BB}\rvert$ denotes the size of the bulletin board, whereas L denotes the maximum number of users allowed in the system. Encryption and decryption take time $O(\lvert S\rvert)$ in all cases. (First two rows are the "naive" constructions with $O(L)$ ciphertext where we encrypt to all public keys in the set.)

In particular, our schemes achieve similar efficiency as the state-of-the-art vanilla (non-distributed) pairing-based broadcast encryption schemes (cf. Table 2).

Our schemes rely on standard assumptions in bilinear groups, such as the bilinear Diffie-Hellman Exponent (BDHE) [4], or the k-Lin assumption [3, 24, 31]. Two of our schemes satisfy strong security guarantees, which ensure that the message is hidden even against an adversary that *adaptively* corrupts honest users and can register malformed public keys to the bulletin board. For the case of the BDHE scheme, this is achieved by a generic transformation that turns any scheme that satisfies a very weak notion of security (that we refer to as semi-selective) to an adaptively secure scheme, following the approach of [21].

Furthermore, we show how to achieve two strengthenings of DBE inspired by recent works on registration-based encryption [16, 17, 22, 25, 12]:

- The first is that of dynamic joins, where users can register public keys and join the bulletin board at any time, and secret keys may require to be updated once a new user joins. We show a generic transformation (following [16, 22, 25, 12]) where users need to check the bulletin board for updates at most $\mathrm{poly}(\log L)$ times throughout the lifetime of the system, while increasing the ciphertext size by a $O(\log L)$ factor.
- The second is that of malicious corruptions, where malicious users can register (possibly-malformed) keys. We show that as long as public keys pass a simple validity check, then the presence of malicious users do not comprise correctness or privacy of our schemes.

1.2 Discussion and Related Works

Prior works. Besides "trivial" schemes, where one simply encrypts the message in parallel for all users in the subsets, the only DBE schemes proposed in the literature are from Wu, Qin, Zhang and Domingo-Ferrer [37] and Boneh and Zhandry [7]. We present an explicit comparison of known DBE schemes, along with their underlying assumptions in Table 1. We mention here that the conference version of [37] presents a construction without a proof of security. On the other hand, the full version [36] presents a *different construction*, along with a security proof. Unfortunately, we show that their argument is flawed by presenting an

attack against their full-version construction (see Appendix A for details). For completeness, we also provide a proof (although only for the weaker notion of selective security) for a slight variant of the conference version [37] in Appendix A.

We shall mention here explicitly that neither work explicitly considered the settings with dynamic joins, and therefore the resulting schemes also have a linear (in L) number of updates. However, we stress that a similar transformation as the one that we present in this work can be applied also to their schemes to reduce the number of updates to $\log L$, while increasing the ciphertext size to $\log L$.

Decentralized broadcast encryption. Phan, Pointcheval, and Strefler [30] put forth the notion of decentralized broadcast encryption, which relies on *interactive* key generation to solve the key-escrow problem in broadcast encryption. That is, upon a new user's arrival, a subset of the users should be online to involve in an interactive protocol with the new user, in order for the keys of the system–new user's key and (part of) the previous ones–to be updated. Furthermore, the ciphertext is not always succinct on the set size, it can vary from 1 to $O(|S|)$, depending on the structure of S. They provided efficient instantiations based on the DDH assumption in pairing-free groups.

In contrast, the model of [37, 7] (which is also the one that we adopt in this work) insist on a *non-interactive* key generation procedure, apart from a single write access to the public bulletin board and up to $O(\log L)$ reads per user and it is required that the ciphertexts are poly-logarithmic in $|S|$ for every possible broadcast-set S.

Registration-based encryption. Another related (but different) notion is that of registration-based encryption [16]. Also in registration-based encryption (RBE) users sample their own secret keys and there is no key authority that knows a master secret key. However RBE assumes the existence of a semi-trusted key curator that aggregates the public keys of the users and generates a short *master public key* that anyone can use to encrypt with respect to identities. On the other hand, in DBE there is no key curator, but just an append-only bulletin board (alternatively, one can think of the key curator as doing nothing). These settings entail two different technical challenges:

- In RBE, the non-triviality of the scheme comes from the size of the master public key, which is required to be sublinear, ideally independent, of the number of users. (No requirement on the size of the ciphertext)
- In DBE, the challenge stems from the size of the ciphertext, which is required to be sublinear, ideally independent, of the size of the set. (No requirement on the size of the master public key)

Because of the above differences the two primitives are formally incomparable, i.e., RBE does not imply DBE and vice-versa. We also point out that DBE (and in fact broadcast encryption) does not support the functionality of encryption "with respect to identity" as in RBE, which is rather a property of *identity-based* broadcast encryption.

Recently Hohenberger et al. introduced the relevant to RBE notion of Registered (ciphertext-policy) attribute-based encryption [25] and, as Agrawal and Yamada [2] observed, CP-ABE implies broadcast encryption. However, the same comparison with RBE applies; R-ABE's objective is to have sublinear master public key, which is orthogonal to sublinear ciphertexts. In fact, their ciphertext is linear in the size of policy, therefore naively using the R-ABE as an DBE yields $O(|S|)$-sized ciphertexts.

User identifiers. We discuss possible choices for assigning user identifiers to participants. For the static settings, where all users are fixed at the beginning of the system, one can simply identify users in the set by their lexicographical ordering. This is what is done in traditional (non-distributed) broadcast encryption systems. In the dynamic settings, where users can post public keys on the bulletin board at different times, one can envision different mechanisms. The simplest one is to identify users by their time of arrival: The bulletin board specifies a counter corresponding to how many users have joined the system so far as well as the identifiers of all the users who have joined the system so far, in order. When a user joins, it checks the counter first, increments it by 1 to j and runs keygen with j and submits her public key, upk_j, with its identifier that also gets appended to the bulletin board; the identifier is now associated with index/time j. When a sender comes along, it looks up the bulletin board to check the identifiers and the corresponding indices, and uses that to determine a set $S \subseteq [L]$. When we say that our scheme supports $\mathrm{poly}(\log L)$ updates, we mean that a decryptor will only need to look up the bulletin board at most $\mathrm{poly}(\log L)$ times for public key updates, even if L users join the system.

Common reference string. We point out explicitly that, similarly to recent works on RBE [22] and R-ABE [25] and R-FE [15], two of our schemes are in the common reference string model, where a reference string must be sampled in such a way that the underlying trapdoor is kept secret. We argue that this assumption is still substantially better than having a trusted authority distributing secret keys for all parties. This is because the common reference string is sampled once and for all and there is no *long term secret* that needs to be stored. Additionally, availability-wise, no trusted authority must remain online to send secret keys and no secure communication channels are needed.

One way to sample a common reference string is to have a one-time ceremony, where a group of non-colluding parties run a multi-party computation protocol to compute the common reference string, each supplying their own randomness. Furthermore, the common reference string can be reused across different instances of the scheme. We also point out that our first scheme has an *updatable* common reference string [23] of the form

$$[\alpha], \ldots, [\alpha^L], [\alpha^{L+2}], \ldots, [\alpha^{2L}]$$

for a secret scalar α. This class of common reference strings can be easily updated using the techniques in [23, 29], where each update can be verified via a simple proof of knowledge of discrete logarithm. This highly mitigates the trust in the common reference string since anyone can update it and even one honest update (the underlying update trapdoor is destroyed) suffices.

Efficiency comparison with non-distributed BE. In Table 2 we provide a comparison of our DBE with traditional BE from bilinear groups. Although we compare primitives with different objectives and functionalities, our objective is to show the cost of getting rid of the trusted authority in broadcast encryption. For fairness, we compare with the variant of our schemes in the *static settings*, i.e., where the set of users is fixed, which is the same as traditional BE. The conclusion is that in comparison to [4] and [21] we achieve essentially the same efficiency properties except for a $\log L$ overhead on the size of the ciphertext (for the variant with efficient updates). [33] achieves the best tradeoff between parameters that we do not achieve. We note that in the case of DBE, due to absence of the trusted authority, we have a quadratic-sized Bulletin Board.

	Assumption	Dist.	Security	|Enc|	|Dec|	|ct|	|BB|	#Updates
BGW05 [4]	BDHE	No	Selective	L	L	3	–	N/A
Scheme 1a	BDHE	Yes	Selective	L	L	3	L^2	L
GW09 [21]	BDHE	No	Adaptive	L	L	6	–	N/A
GKW18 [19]	SXDH	No	Adaptive	L	L	4	–	N/A
Scheme 1b	BDHE	Yes	Adaptive	L	L	6	L^2	L
Scheme 2	SXDH	Yes	Adaptive	L	L	4	L^2	L

Table 2: Comparison with existing BE schemes with $O(1)$-size ciphertexts from Bilinear Groups. The notation in the table ignores constants and factors that depend only on the security parameters, except for $|\mathsf{ct}|$ which is concrete in number group elements. $|\mathsf{Enc}|$ denotes the overall size of information needed to encrypt a message (e.g. mpk or crs), and $|\mathsf{Dec}|$ the overall size of information to decrypt (e.g. usk_j or sk). $|\mathsf{BB}|$ denotes the size of the bulletin board (for DBE), whereas L denotes the maximum number of users allowed in the system. We omit from the comparison schemes like [33] which achieves $O(N^{1/3})$ parameters but not $O(1)$-size ciphertexts. Recall SXDH = 1-Lin.

1.3 Open Problems

We view our work as opening a promising new line of research in pairing-based broadcast encryption. In fact there is a number of problems that our work leaves open: For example, we ask what are the optimal parameters for pairing-based DBE, and whether we can achieve similar tradeoffs as for the case of vanilla broadcast encryption [33]. Furthermore, an outstanding open problem is whether we can construct distributed broadcast encryption from other assumptions, such as lattice-based computational problems.

2 Technical Overview

2.1 High-level overview

Syntax for distributed broadcast encryption. We begin with the syntax for distributed broadcast encryption in the simplest setting. There are L users in the system and a public parameter pp given to all users.

- User j given pp, generates a public/private key pair $(\mathsf{upk}_j, \mathsf{usk}_j)$ and posts upk_j to a public bulletin board.
- Encryption to a set S takes as input $\{\mathsf{upk}_j\}_{j\in S}$ and a message M and outputs a ciphertext ct.
- Decryption takes as input a set S, a ciphertext ct, the public keys $\{\mathsf{upk}_j\}_{j\in S}$, and a secret key usk_i for some $i \in S$.

The basic semantic security requirement says that given an encryption ct of M for any set S, along with $\{\mathsf{upk}_j\}_{j\in[N]}$ and $\{\mathsf{usk}_j\}_{j\notin S}$, the message M remains hidden. The stronger notion of adaptive security (where the adversary can choose S after seeing some of the public keys) can be achieved with a semi-generic transformation due to [21], which we adapt to the distributed setting. However, to keep things simple, in this overview we focus on selective security, where the set S is fixed in advance.

Looking ahead, we present two constructions of DBE satisfying these basic requirements, one based on DBHE and the second based on k-Lin. These two schemes constitute the basic building blocks for our "full

fledged"–enhanced with additional requirements described below–DBE schemes. We defer an overview of both constructions for now, and proceed instead to describe the additional requirements for DBE and how we achieve them via a series of generic transformations.

Malicious corruptions. Next, we strengthen our security requirements to allow for malicious corruptions, where a user j controlled by an adversary may post an arbitrary, malformed upk_j^* to the bulletin board, as long as upk_j^* passes some validity check. We have two requirements: first, we want functionality to hold even amidst malicious corruptions, namely an honest user i should correctly decrypt an honestly generated encryption for a set S where $i \in S$, even if S contains malformed public keys. Next, we require that semantic security holds even if all keys outside S (i.e., $[L] \setminus S$) are corrupted. We note that, as typically in Broadcast Encryption, it's meaningless to assume corruptions inside S; an adversary controlling a user inside S can trivially decrypt.

We show that any scheme that is semantically secure without malicious corruptions is also semantically secure with malicious corruptions. The reduction is fairly straight-forward, crucially relying on the fact that our syntax for encryption takes as input only the public keys of users in S, and since we do not allow malicious corruptions inside S, malicious coruptions do not affect the distribution of the challenge ciphertext.

Reducing the number of updates. It is typical in dynamic settings without a private key generator authority that the users have to update their (decryption keys). Following similar settings, such as Registration-based Encryption, we put forward a generic transformation to any DBE scheme that allows users to update their decryption keys only logarithmic number—in the total number of users, L—times throughout the history of the system. We defer to section 4.3 for the technical details.

2.2 Scheme from BDHE

We rely on an asymmetric bilinear group $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$ of prime order p where $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. We use $[\cdot]_1, [\cdot]_2, [\cdot]_T$ to denote component-wise exponentiations in respective groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$. For this overview we implicitly assume that all algorithms take the description of the group (together with corresponding random generators) as input, $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, [1]_1, [1]_2, e)$.

BGW Broadcast Encryption. Our starting point is the BGW broadcast encryption scheme [4] which we recap below:

$$
\begin{aligned}
\mathsf{pp} &= \left([\alpha]_1, \ldots, [\alpha^L]_1, [\alpha]_2, \ldots, [\alpha^L]_2, [\alpha^{L+2}]_2, \ldots, [\alpha^{2L}]_2\right) \\
\mathsf{msk} &= t, && t \leftarrow_\$ \mathbb{Z}_p^* \\
\mathsf{mpk} &= ([t]_1) \\
\mathsf{ct} &= \left([s]_1, [s(t + \sum_{j \in S} \alpha^j)]_1, [s\alpha^{L+1}]_T \cdot M \right), && s \leftarrow_\$ \mathbb{Z}_p^* \\
\mathsf{sk}_i &= [t\alpha^{L+1-i}]_2
\end{aligned}
$$

Decryption by a user $i \in S$ is based on the following equation:

$$e\left([s(t+\sum_{j\in S}\alpha^j)]_1, [\alpha^{L+1-i}]_2\right) =$$
$$= e\left([s]_1, \overbrace{\boxed{[t\alpha^{L+1-i}]_2}}^{\mathsf{sk}_i} \cdot [\sum_{j\in S, j\neq i} \alpha^{L+1+j-i}]_2\right) [s\alpha^{L+1}]_T$$

Observe that the secret keys of all users depend on the same secret value: the master secret key t. It is worth noting that this is a technique that is common even among all subsequent broadcast encryption schemes. The natural question when adapting BGW to the distributed setting is: who chooses t?

Our DBE. Our core technique is the following: let t be $t = \sum_{i\in S} t_i$ where t_i is chosen by the i'th user. This transforms the above decryption equation to:

$$e\left([s\sum_{i\in S}(t_j+\alpha^j)]_1, [\alpha^{L+1-i}]_2\right) =$$
$$= e\left([t_i\alpha^{L+1-i} + \sum_{j\in S, j\neq i}\left(t_j\alpha^{L+1-i} + \alpha^{L+1+j-i}\right)]_1, [s]_2\right)[s\alpha^{L+1}]_T =$$
$$= e\left(\overbrace{\boxed{[t_i\alpha^{L+1-i}]_1}}^{\mathsf{sk}_{i,i}} \prod_{j\in S, j\neq i} \left\{\overbrace{\boxed{[t_j\alpha^{L+1-i}]_1}}^{\mathsf{sk}_{i,j}} \cdot [\alpha^{L+1+j-i}]_1\right\}, [s]_2\right)[s\alpha^{L+1}]_T$$

Now, as it is evident, the cross-terms $t_j\alpha^{L+1-i}$ appear in the decryption equation. Therefore, in order to make decryption possible, it is inevitable that the decryptor i knows these terms.

This affects the efficiency of the scheme (making the user's necessary information to decrypt linear in L), but is very convenient for the distributed setting. The intuition is that each user i can be attached to one distinct t_i that she can sample locally. Then user i publishes the cross-terms of all the other users $\mathsf{sk}_{i,j} = t_i\alpha^{L+1-j}$, except of course for $\mathsf{sk}_{i,i} = t_i\alpha^{L+1-i}$ which is her secret key. In terms of correctness, observe that $\mathsf{sk}_{i,i}$ is only used in the decryption equation of i, i.e. it never plays the role of a cross-term. Additionally, publishing $\mathsf{sk}_{i,j}$ doesn't affect security as long as $\mathsf{sk}_{i,i}$ remains secret.

To summarize, our DBE is given as follows:

$$\begin{aligned}
\mathsf{pp} &= \left([\alpha]_1, \ldots, [\alpha^L]_1, [\alpha]_2, \ldots, [\alpha^L]_2, [\alpha^{L+2}]_2, \ldots, [\alpha^{2L}]_2\right) \\
\mathsf{usk}_j &= [t_j\alpha^{L+1-j}]_2, && t_j \leftarrow_\$ \mathbb{Z}_p^* \\
\mathsf{upk}_j &= \left([t_j]_1, [t_j\alpha]_2, \ldots, [t_j\alpha^{L+1-j-1}]_2, [t_j\alpha^{L+1-j+1}]_2, \ldots, [t_j\alpha^L]_2\right) \\
\mathsf{ct} &= \left([s]_1, [s(\sum_{j\in S} t_j + \alpha^j)]_1, [s\alpha^{L+1}]_T \cdot M\right), && s \leftarrow_\$ \mathbb{Z}_p^*
\end{aligned}$$

It is worth noting that in terms of efficiency the public parameters and the ciphertext remain unaffected. Futhermore, from the point of view of the encryptor the (asymptotic) storage overhead is the same as BGW: In BGW (translated to asymmetric groups) an encryptor needs $\mathsf{pp} = (\{[\alpha^i]_1\}_{i\in[L]}, \{[\alpha^i]_2\}_{i\in[2L]\setminus\{L+1\}})$ and $[t]_1$ while in our case the same pp and $[t_1]_1, \ldots, [t_L]_1$, therefore both are $O(L)$.

Security Proof Sketch. The decision Blinear Diffie-Hellman Exponent assumption [4] says that $[s\alpha^{L+1}]_T$ is pseudorandom given $(\{[\alpha^i]_1\}_{i\in[L]}, \{[\alpha^i]_2\}_{i\in[2L]\setminus\{L+1\}}) := \mathsf{pp}$ and $[s]_1 := \mathsf{ct}_1$. The security reduction for selective security given S^* proceeds as follows:

- for all $j \notin S^*$, reduction samples t_j "in the clear";
- for all $j \in S^*$, reduction samples $\tilde{t}_j$ and implicitly sets $t_j := \tilde{t}_j - \alpha^j$.

Now, observe that

$$\mathsf{ct} = \left([s]_1, [s \sum_{j\in S} \tilde{t}_j]_1, [s\alpha^{L+1}]_T \cdot M \right)$$

the reduction can simulate:

- the public parameters using the input of the assumption;
- the challenge ct given $[s]$ and the target of the assumption.
- terms $[t_j\alpha^{L+1-i}]_2, j \notin S^*$ and any i, we can simulate using t_j
- terms $[t_j\alpha^{L+1-i}]_2, j \in S^*, i \neq j$: we can simulate using $\tilde{t}_j$ and $[\alpha^{L+1-i+j}]_2$
- terms $[t_j\alpha^{L+1-j}]_2, j \in S^*$: appear only in sk_j, which the adversary is not allowed to query

Interestingly, our modification described above allows us to prove our DBE scheme semi-statically secure, something that is not possible for BGW. Semi-static security is a strengthening of static security in the sense that the adversary still outputs the target set S^* a-priori but at the challenge stage is allowed to ask a ciphertext for any subset $S^{**} \subseteq S^*$. This security notion is interesting since Gentry and Waters showed a generic transformation (in the Random Oracle model) from a semi-statically secure scheme to fully adaptive [21]. We adapt this transformation in the distributed broadcast encryption setting in section 4.2.

2.3 Scheme from k-Lin

Our k-Lin Distributed Broadcast Encryption conceptually follows the framework of Gay et al. [19] that construct (plain) Broadcast Encryption from k-Lin. The [19] scheme itself conceptually holds similarities with the Broadcast Encryption scheme of Gentry and Waters (GW) [21]. In order to ease the presentation, first we give some context to the reader on the GW construction.

GW Broadcast Encryption. The GW scheme works as follows:

$$\begin{aligned}
\mathsf{pp} &= \left([w_1]_1, \ldots, [w_L]_1\right), w_i \leftarrow \mathbb{Z}_p^* \\
\mathsf{msk} &= \left([\alpha]_2, [w_1]_2, \ldots, [w_L]_2\right), \alpha \leftarrow_{\$} \mathbb{Z}_p^* \\
\mathsf{mpk} &= [\alpha]_T \\
\mathsf{ct} &= \left([s]_1, [s\sum_{j\in S} w_j]_1, [s\alpha]_T \cdot M\right), s \leftarrow \mathbb{Z}_p^* \\
\mathsf{sk}_i &= \left([\alpha + w_i r_i]_2, [r_i]_2, \{[r_i w_j]_2\}_{j\in[L]\setminus\{i\}}\right), r_i \leftarrow \mathbb{Z}_p^*
\end{aligned}$$

Decryption uses the fact that if $i \in S$, then:

$$e\left([s]_1, \overbrace{\boxed{[\alpha + r_i w_i]_2}}^{\mathsf{sk}_{i,i}} \cdot \prod_{j\in S, j\neq i} \overbrace{\boxed{[r_i w_j]_2}}^{\mathsf{sk}_{i,j}}\right) = e\left([s\sum_{i\in S} w_j]_1, [r_i]_2\right)[s\alpha]_T$$

GKW Broadcast Encryption. The observation of Gay et al. [19] is that one can 'push' $\mathsf{sk}_{i,j}$ to the parameters (sampling a-priori all r_i's). This makes the public parameters quadratic, but achieves constant-sized decryption keys. This gives us:

$$\begin{aligned}
\mathsf{pp} &= \left(\{[w_i]_1\}_{i\in[L]}, \{[r_i]_2\}_{i\in[L]}, \{[r_i w_j]_2\}_{i,j\in[L], i\neq j}\right), w_i, r_i \leftarrow \mathbb{Z}_p^* \\
\mathsf{msk} &= \left([\alpha]_2, [w_1]_2, \ldots, [w_L]_2, r_1, \ldots, r_L\right) \\
\mathsf{mpk} &= [\alpha]_T \\
\mathsf{ct} &= \left([s]_1, [s\sum_{j\in S} w_j]_1, [s\alpha]_T \cdot M\right), s \leftarrow \mathbb{Z}_p^* \\
\mathsf{sk}_i &= [\alpha + w_i r_i]_2
\end{aligned}$$

Looking ahead, GKW (almost) generically transforms the scheme to the k-Lin setting to achieve adaptive security (without the use of random oracles). For presentational purposes we postpone this to the end of the section and continue with the traditional setting, stated above.

Towards our DBE. In the distributed case we crucially rely on the fact that $\mathsf{sk}_{i,j}$'s can be 'pushed' to the public parameters, since there is no private key generator authority to sample r_i on-the-fly to extract the decryption key of the user i.

However, we're not done, there still remains the natural question: who samples α, or in more general, how are the secret keys sk_i formed so that they do not depend on a (universal) master secret key? We resolve this as follows: As in our previous DBE scheme, user i samples a fresh $t_i \leftarrow \mathbb{Z}_p^*$. Then, we replace w_j in ct with $t_j + w_j$, which yields

$$\mathsf{ct} = \left([s]_1, [s\sum_{j\in S}(t_j + w_j)]_1, [s\alpha]_T \cdot M\right)$$

Doing the same with sk_i and the corresponding entries of the public parameters (the GW parts of sk_i that were 'pushed' to the pp) yields:

$$\mathsf{sk}_i = [\alpha + (w_i + t_i)r_i]_2$$
$$\mathsf{pp}^{(3)} = \{[(t_j + w_j)r_i]_2\}_{i,j\in[L],i\neq j}$$

This leads us naturally to the following DBE scheme:

$$\begin{aligned}
\mathsf{pp} &= \left([\alpha]_T, \{[w_i]_1\}_{i\in[L]}, \{[r_i]_2\}_{i\in[L]}, \{[\alpha + r_i w_i]_2\}_{i\in[L]}, \{[r_i w_j]_2\}_{i,j\in[L],i\neq j}\right) \\
\mathsf{usk}_j &= [t_j r_j]_2, \quad t_j \leftarrow \mathbb{Z}_p^* \\
\mathsf{upk}_j &= ([t_j]_1, [t_j r_1]_2, \ldots, [t_j r_{j-1}]_2, [t_j r_{j+1}]_2, \ldots, [t_j r_L]_2) \\
\mathsf{ct} &= \left([s]_1, [s\sum_{j\in S}(t_j + w_j)]_1, [s\alpha]_T \cdot M\right) \; s \leftarrow \mathbb{Z}_p^*
\end{aligned}$$

Regarding efficiency, the above Distributed Broadcast Encryption Scheme preserves the decryption and encryption key sizes, $O(L)$, of the GW scheme, but at the cost of having quadratic public parameters instead of linear.

Security Intuition. To gain some intuition about security, consider an adversary that tries to recover $[s\alpha]_T$ by computing $e([s\sum_{j\in S}(t_j + w_j)]_1, [r_i]_2)$ and $e([s], [\alpha + w_i r_i])$. Then notice that:

- if $i \notin S$, it can't cancel out $[sw_i r_i]_T$.
- if $i \in S$, it can't cancel out $[st_i r_i]_T$ without usk_i, which of course is not allowed to query in the security game of a (distributed) broadcast encryption (otherwise it trivially recovers the message).

Final Scheme from k-Lin. To obtain a scheme under k-Lin following [11, 19], we sample $\mathbf{A} \leftarrow \mathbb{Z}_p^{k\times(k+1)}$ and make the following substitutions:

$$\begin{aligned}
&s \mapsto \mathbf{s}^\top\mathbf{A} \in \mathbb{Z}_p^{1\times(k+1)}, \; \alpha \mapsto \mathbf{k} \in \mathbb{Z}_p^{k+1}, \\
&w_j \mapsto \mathbf{W}_j \in \mathbb{Z}_q^{(k+1)\times k}, \\
&t_j \mapsto \mathbf{T}_j \in \mathbb{Z}_q^{(k+1)\times k}, \\
&[w_j]_1, [\alpha]_T \mapsto [\mathbf{A}\mathbf{W}_j]_1, [\mathbf{A}\mathbf{k}]_T
\end{aligned}$$

We defer further details to Section 6.

3 Preliminaries

Notation We write λ for the security parameter. By $[N]$ we denote the set of integers $\{1, \ldots, N\}$ and, more generally, by $[A, B]$ the set $\{A, \ldots, B\}$ for any $A, B \in \mathbb{Z}, A \leq B$. With $x \leftarrow_\$ X$ we denote the procedure of x being uniformly sampled from a finite set X. Throughout our work "PPT" stands for probabilistic polynomial-time algorithm.

3.1 Bilinear Groups

A generator $\mathcal{BG}$ takes as input a security parameter 1^λ and outputs a description $\mathbb{G} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$, where p is a prime of $\Theta(\lambda)$ bits, $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ are cyclic groups of order p, and $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is a non-degenerate bilinear map. We require that the group operations in $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$ and the bilinear map e are computable in deterministic polynomial time in λ. Let $g_1 \in \mathbb{G}_1$, $g_2 \in \mathbb{G}_2$ and $g_T = e(g_1, g_2) \in \mathbb{G}_T$ be the respective generators. We employ the *implicit representation* of group elements: for a matrix $\mathbf{M}$ over $\mathbb{Z}_p$, we define $[\mathbf{M}]_1 := g_1^{\mathbf{M}}, [\mathbf{M}]_2 := g_2^{\mathbf{M}}, [\mathbf{M}]_T := g_T^{\mathbf{M}}$, where exponentiation is carried out component-wise. We denote $[A]_s \cdot [B]_s = [A + B]_s$ for $s = 0, 1, T$. We further define, for any $\mathbf{A} \in \mathbb{Z}_p^{n\times m}$ and $\mathbf{B} \in \mathbb{Z}_{m\times \ell}$,

$$e([\mathbf{A}]_1, [\mathbf{B}]_2) := [\mathbf{AB}]_T \in \mathbb{G}_T^{n\times \ell}.$$

We recall two standard assumptions over Bilinear Groups, that we will use in the following sections. First, the (decision) Bilinear Diffie-Hellman Exponent assumption introduced by Boneh et. al. [4].

Assumption 3.1 (Decision BDHE assumption)**.** *Let $\mathcal{BG}$ be a bilinear group generator,* $\mathsf{bg} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, [1]_1, [1]_2, e) \leftarrow_\$ \mathcal{BG}(1^\lambda)$*, $\alpha, s \leftarrow_\$ \mathbb{Z}_p^*$ and define*

$$\mathcal{D} = \left(\left\{[\alpha^j]_1\right\}_{j\in[q]}, \left\{[\alpha^j]_2\right\}_{j\in[2q],j\neq q+1}, [s]_1\right)$$

and $T \leftarrow_\$ \mathbb{G}_T$. Then for any PPT adversary $\mathcal{A}$ and any positive integer q:

$$\mathbf{Adv}^{\mathsf{dBDHE}}_{\mathcal{BG},q,\mathcal{A}}(\lambda) := \left|\Pr[\mathcal{A}\left(\mathsf{bg}, \mathcal{D}, [s\alpha^{q+1}]_T\right) = 1] - \Pr[\mathcal{A}\left(\mathsf{bg}, \mathcal{D}, T\right) = 1]\right| = \mathrm{negl}(\lambda)$$

where the above probabilities are taken over the choices of bg, α, s *and* T*.*

We also recall the k-Lin assumption [3, 24, 31] which belongs to the more general family of Matrix Diffie-Hellman Assumptions [13]. Define the distribution $\mathcal{L}_k$ outputting a matrix $\mathbf{A} \in \mathbb{Z}_p^{(k+1)\times k}$ as:

$$\mathbf{A} = \begin{pmatrix} a_1 & 0 & \dots & 0 \\ 0 & a_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & a_k \\ 1 & 1 & \dots & 1 \end{pmatrix}$$

where $a_1, \dots, a_k \leftarrow_\$ \mathbb{Z}_p$. The k-Lin assumption is stated below.

Assumption 3.2 (k-Lin assumption)**.** *Let $\mathcal{BG}$ be a bilinear group generator,* $\mathsf{bg} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, [1]_1, [1]_2, e) \leftarrow_\$ \mathcal{BG}(1^\lambda)$*, k any positive integer, $\mathbf{v} \leftarrow_\$ \mathbb{Z}_p^k$, $\mathbf{u} \leftarrow_\$ \mathbb{Z}_p^{k+1}$ and $\mathbf{A} \leftarrow_\$ \mathcal{L}_k$ Then for any PPT adversary $\mathcal{A}$:*

$$\mathbf{Adv}^{k\text{-}Lin}_{\mathcal{BG},q,\mathcal{A}}(\lambda) := |\Pr[\mathcal{A}\left(\mathsf{bg}, [\mathbf{A}]_s, [\mathbf{Av}]_s\right) = 1] - \Pr[\mathcal{A}\left(\mathsf{bg}, [\mathbf{A}]_s, [\mathbf{u}]_s\right) = 1]| = \mathrm{negl}(\lambda)$$

where the above probabilities are taken over the choices of $\mathsf{bg}, \mathbf{A}, \mathbf{v}, \mathbf{u}$*.*

We note that if $s = 1$ we have the k-Lin assumption for group $\mathbb{G}_1$ and similarly $s = 2$ for $\mathbb{G}_2$.

4 Definitions

We consider a model where the system is initialized with some public parameters, given an upper bound on the number of users, L. Users are indexed by a unique identifier $j \in [L]$ which can be, e.g., their time of arrival. Upon each user joining the system, they append their public key on a bulletin board that we assume all users in the system have access to. Importantly, the bulletin board is append-only, and we implicitly assume that all parties involved scrutinize that the public keys of the other parties are well-formed. If not, the public key is simply discarded. By looking at the public parameters and at the users public keys, it is then possible to encrypt a message for a subset $S \subseteq [L]$ of the public keys. What makes the scheme non-trivial is the fact that the size of the ciphertext must be sublinear (and ideally independent) of the size of S.

4.1 Distributed Broadcast Encryption

We present the formal definitions in the following. The syntax is canonical for broadcast encryption, except that each user samples locally a key pair $(\mathsf{usk}, \mathsf{upk})$ and, consequently, the encryption and decryption will take as input the public keys corresponding to the users in the set S. For starters, we define a minimal notion of security, where the adversary is allowed to choose a set S^* for the challenge ciphertext at the beginning of the experiment, but it is otherwise not allowed to corrupt honest users. Later in this Section, we will show how to generically lift this definition to the more challenging settings with adaptive corruptions.

To prevent the adversary from registering public keys that would sabotage the decryption of honest users, we define a validity-check predicate that ensures that the keys are well-formed. Then the correctness guarantee is that, if the check succeeds, then correctness must hold unconditionally.

Definition 4.1 (Distributed Broadcast Encryption)**.** *Let λ be a security parameter and let $\mathbb{M} = \{\mathbb{M}_\lambda\}_{\lambda\in\mathbb{N}}$ be the message space. A distributed broadcast encryption scheme with message space $\mathbb{M}$ is a tuple of efficient algorithms $\Pi_{\mathsf{DBE}} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ with the following properties:*

- $\mathsf{Setup}(1^\lambda, 1^L) \to \mathsf{pp}$*: On input the security parameter λ and the number of slots L, the setup algorithm returns some public parameters* pp*.*
- $\mathsf{KeyGen}(\mathsf{pp}, j) \to (\mathsf{usk}_j, \mathsf{upk}_j)$*: On input the public parameters and a slot index $j \in [L]$, the key generation algorithm generates a secret key* usk_j *and a public key* upk_j *for the j-th slot.*
- $\mathsf{Enc}(\mathsf{pp}, \{\mathsf{upk}_j\}_{j\in S}, S, M) \to \mathsf{ct}$*: On input the public parameters* pp*, the public keys* $\{\mathsf{upk}_j\}_{j\in S}$*, a subset $S \subseteq [L]$, and a message $M \in \mathbb{M}$, the encryption algorithm returns a ciphertext* ct*.*
- $\mathsf{Dec}(\mathsf{pp}, \{\mathsf{upk}_j\}_{j\in S}, \mathsf{usk}_i, \mathsf{ct}, S, i) \to M$*: On input the public parameters* pp*, the public keys* $\{\mathsf{upk}_j\}_{j\in S}$*, a secret key* usk_i *a ciphertext* ct*, a subset S, and an index i, the decryption algorithm returns a message M.*

Moreover, the algorithms must satisfy the following properties.

- ***Correctness:*** *For all $\lambda \in \mathbb{N}$, $L \in \mathbb{N}$, $j \in [L]$, all* pp *in the support of* $\mathsf{Setup}(1^\lambda, 1^L)$*, all* $(\mathsf{usk}_i, \mathsf{upk}_i)$ *in the support of* $\mathsf{KeyGen}(\mathsf{pp}, i)$*, all* $\{\mathsf{upk}_j\}_{j\neq i}$ *such that* $\mathsf{isValid}(\mathsf{pp}, \mathsf{upk}_j, j) = 1$*, all $M \in \mathbb{M}$, all $S \subseteq [L]$ such that $i \in S$, it holds that*

$$\Pr\left[\mathsf{Dec}(\mathsf{pp}, \{\mathsf{upk}_j\}_{j\in S}, \mathsf{usk}_i, \mathsf{ct}, S, i) = M : \mathsf{Enc}(\mathsf{pp}, \{\mathsf{upk}_j\}_{j\in S}, S, M)\right] = 1.$$

- ***Verifiable Keys:*** *There exists an efficient algorithm* $\mathsf{isValid}$ *such that for all* $\lambda \in \mathbb{N}$, $L \in \mathbb{N}$, $j \in [L]$, *all* pp *in the support of* $\mathsf{Setup}(1^\lambda, 1^L)$*, it holds that*

$$(\cdot, \mathsf{upk}_j) \in \mathsf{KeyGen}(\mathsf{pp}, j) \implies \mathsf{isValid}(\mathsf{pp}, \mathsf{upk}_j^*, j) = 1$$

- ***(Selective) Security:*** *Define the following experiment between an adversary* $\mathcal{A}$ *and a challenger, parameterized by a bit* b*.*
  - ***Setup Phase:*** *The adversary* $\mathcal{A}$ *sends a set* $S^* \subseteq [L]$ *to the challenger, who samples* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^L)$ *and gives* pp *to* $\mathcal{A}$*.*
  - ***Key Generation Phase:*** *For all* $j \in S^*$*, the challenger samples a key pair* $(\mathsf{usk}_j, \mathsf{upk}_j) \leftarrow \mathsf{KeyGen}(\mathsf{pp}, j)$ *and sends* $\{\mathsf{upk}_j\}_{j \in S^*}$ *to* $\mathcal{A}$*.*
  - ***Challenge Phase:*** *The adversary sends a pair of messages* $M_0^*, M_1^* \in \mathbb{M}$*. The challenger computes*

$$\mathsf{ct}^* \leftarrow \mathsf{Enc}(\mathsf{pp}, \{\mathsf{upk}_j\}_{j \in S^*}, S^*, M_b^*)$$

    and sends it over to $\mathcal{A}$*.*
  - ***Output Phase:*** *At the end of the experiment, algorithm* $\mathcal{A}$ *outputs a bit* $b' \in \{0, 1\}$*, which is the output of the experiment.*

  We say that a distributed broadcast encryption scheme Π_{dRBE} *is selectively secure if for all polynomials* $L = L(\lambda)$ *and all efficient adversaries* $\mathcal{A}$*, there exists a negligible function* negl *such that for all* $\lambda \in \mathbb{N}$*:*

$$\left| \Pr\left[b = b'\right] - \frac{1}{2} \right| = \mathrm{negl}(\lambda).$$

4.2 Semi-Selective to Adaptive

In the following we show a series of transformation that allow us to compile a scheme that satisfies a weak notion of security into one that satisfies adaptive security. We proceed in three steps:

- First, we define a stronger notion of selective security, called *semi-selective* security.
- Second, we show how the transformation described in [21] allows us to build a scheme with adaptive security, albeit only for honestly generated keys.
- Third we show how to handle maliciously generated keys.

Let us first define the notion of *semi-selective* security. The experiment is unchanged, except that the adversary in the challenge phase can specify any $S^{**} \subseteq S^*$ and the challenge ciphertext is defined as

$$\mathsf{ct}^* \leftarrow \mathsf{Enc}(\mathsf{pp}, \{\mathsf{upk}_j\}_{j \in S^{**}}, S^{**}, M_b^*).$$

From Semi-Selective to Passive-Adaptive. Before showing the generic transformation, we provide the formal definition of passive-adaptive security below.

Definition 4.2 (Passive-Adaptive Security)**.** *Define the following experiment between an adversary* $\mathcal{A}$ *and a challenger, parameterized by a bit* b*.*

- ***Setup Phase:*** *The challenger samples* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^L)$ *and gives* pp *to* $\mathcal{A}$.
- ***Key Query Phase:*** *The adversary* $\mathcal{A}$ *is provided with access to the following oracles.*
  - ***Key Generation Oracle:*** *On input an index* $j \in [L]$*, check if this index was queried before to this oracle, and return* $\perp$ *if this is the case. Otherwise, sample* $(\mathsf{usk}_j, \mathsf{upk}_j) \leftarrow \mathsf{KeyGen}(\mathsf{pp}, j)$ *and sent* upk_j *to* $\mathcal{A}$.
  - ***Key Corrupt Oracle:*** *On input an index* $j \in [L]$*, check if* upk_j *was sampled in the key generation oracle and return the corresponding* usk_j *to* $\mathcal{A}$*. In this case, we refer to the index* j *as "corrupted". Otherwise, return* $\perp$.
- ***Challenge Phase:*** *The adversary sends a pair of messages* $M_0^*, M_1^* \in \mathbb{M}$ *and a set* S^**. If there exists some index* $j \in S^*$ *such that* j *was corrupted, then the challenger aborts the experiment. Otherwise it computes*
$$\mathsf{ct}^* \leftarrow \mathsf{Enc}(\mathsf{pp}, \{\mathsf{upk}_j\}_{j \in S^*}, S^*, M_b^*)$$
and sends it over to $\mathcal{A}$.
- ***Output Phase:*** *At the end of the experiment, algorithm* $\mathcal{A}$ *outputs a bit* $b' \in \{0,1\}$*, which is the output of the experiment.*

We say that a distributed broadcast encryption scheme Π_{dRBE} *is passive-adaptively secure if for all polynomials* $L = L(\lambda)$ *and all efficient adversaries* $\mathcal{A}$*, there exists a negligible function* negl *such that for all* $\lambda \in \mathbb{N}$*:*
$$\left| \Pr\left[b = b'\right] - \frac{1}{2} \right| = \mathrm{negl}(\lambda).$$

We are now ready to state the intermediate result, which follows immediately by a transformation from [21]. Although this transformation was originally presented in the context of (non-distributed) broadcast encryption, we observe that the same proof strategy works here.

Lemma 4.3 (Semi-Selective to Passive-Adaptive)**.** *Let* Π_{DBE} *be a semi-selectively secure distributed broadcast encryption scheme. Then there exists a scheme* Π'_{DBE} *that is passive-adaptively secure.*

Proof Sketch. The transformation is identical to [21] and we sketch it here only for completeness.

- Setup: Run the setup of the original scheme except that we double the number of users $2L$.
- Key Generation: On input an index $j \in [L]$, the user generates two keys
$$(\mathsf{usk}_{2j}, \mathsf{upk}_{2j}) \leftarrow \mathsf{KeyGen}(\mathsf{pp}, 2j) \text{ and } (\mathsf{usk}_{2j-1}, \mathsf{upk}_{2j-1}) \leftarrow \mathsf{KeyGen}(\mathsf{pp}, 2j-1)$$
then it flips a coin and throws away one of the two secret keys.
- Encryption: For all $j \in S$, sample a random bit t_j, then define two sets
$$S_0 = \{2j - t_j\}_{j \in S} \text{ and } S_1 = \{2j - (1 - t_j)\}_{j \in S}.$$
Encrypt the message M with respect to both S_0 and S_1 and attach $\{t_j\}_{j \in S}$ to the ciphertext.
- Decryption: Decrypt one of the two ciphertexts, depending on which secret key was kept.

Remark 4.4. *Observe that the compiled scheme has public parameters and ciphertext doubled, when compared to the original scheme. Furthermore, the ciphertext is augmented with a string of $|S|$ bits. As already observed in [21], this can be reduced to λ bits in the random oracle model.*

Next, we show that the scheme is passive-adaptively secure. The proof proceeds in two steps, first we switch the encryption of under the set S_0 to encrypt a random message and then the encryption under S_1. Since the argument is identical, we only outline the reduction for the first case.

- Sample a random string $T = (t_1, \ldots, t_L) \leftarrow \{0,1\}^L$. Set $S^* = \{2j - t_j\}_{j \in [L]}$ as the initial set for the semi-selective security.
- Receive pp from the challenger, along with the keys $\{\mathsf{upk}_j^*\}_{j \in S^*}$.
- Activate the passive-adaptive adversary on input pp.
- Answer the queries of the adversary on index j as follows:
  - Key Generation Oracle: On input an index j, set $\mathsf{upk}_{2j-t_j} = \mathsf{upk}_j^*$ and sample $\mathsf{upk}_{2j-(1-t_j)}$ with the knowledge of the secret key.
  - Key Corrupt Oracle: On input an index j, check if a key was created for this index and return $\mathsf{usk}_{2j-(1-t_j)}$ in case.
- In the challenge phase, the adaptive adversary specifies a set $S^{**} \subseteq [L]$, which must be a subset of the S^* and furthermore must not include any corrupted index. The reduction sets Set $S_0 = \{2i - t_i\}_{i \in S^{**}}$ and send S_0 to the challenger, who responds with ct^*. Then it samples the other ciphertext honestly for the set $S_1 = \{2i - (1 - t_i)\}_{i \in S^{**}}$ and sends both to the adversary.
- In the output phase, return whatever the adversary returns.

Observe that the reduction perfectly simulates the view of the adversary and therefore they have the same advantage. □

From Passive to Active Security. Next, we show that any scheme that satisfies passive-adaptive security also satisfies active-adaptive security, which we define below. The difference with respect to the previous notion, is that we now allow the adversary to register malicious keys, captured by an additional "Malicious Corrupt Oracle".

Definition 4.5 (Active-Adaptive Security)**.** *Define the following experiment between an adversary $\mathcal{A}$ and a challenger, parameterized by a bit b.*

- ***Setup Phase:*** *The challenger samples $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^L)$ and gives pp to $\mathcal{A}$.*
- ***Key Query Phase:*** *The adversary $\mathcal{A}$ is provided with access to the following oracles.*
  - ***Key Generation Oracle:*** *On input an index $j \in [L]$, check if this index was queried before to this oracle or to the malicious corrupt oracle, and return $\perp$ if this is the case. Otherwise sample $(\mathsf{usk}_j, \mathsf{upk}_j) \leftarrow \mathsf{KeyGen}(\mathsf{pp}, j)$ and send upk_j to $\mathcal{A}$.*
  - ***Key Corrupt Oracle:*** *On input an index $j \in [L]$, check if upk_j was sampled in the key generation oracle and return the corresponding usk_j to $\mathcal{A}$. In this case, we refer to the index j as "corrupted". Otherwise, return $\perp$.*

– ***Malicious Corrupt Oracle:*** *On input* (j, upk_j^*) *check if this index* $j \in [L]$ *was queried before to this oracle or to the key generation oracle, and return* $\perp$ *if this is the case. Otherwise Store* upk_j^* *and label the index* j *as "corrupted".*

- ***Challenge Phase:*** *The adversary sends a pair of messages* $M_0^*, M_1^* \in \mathbb{M}$ *and a set* S^**. If there exists some index* $j \in S^*$ *such that* j *was corrupted, then the challenger aborts the experiment. Otherwise it computes*

$$\mathsf{ct}^* \leftarrow \mathsf{Enc}(\mathsf{pp}, \{\mathsf{upk}_j\}_{j \in S^*}, S^*, M_b^*)$$

and sends it over to $\mathcal{A}$*.*

- ***Output Phase:*** *At the end of the experiment, algorithm* $\mathcal{A}$ *outputs a bit* $b' \in \{0, 1\}$*, which is the output of the experiment.*

We say that a distributed broadcast encryption scheme Π_{dRBE} *is adaptively secure if for all polynomials* $L = L(\lambda)$ *and all efficient adversaries* $\mathcal{A}$*, there exists a negligible function* negl *such that for all* $\lambda \in \mathbb{N}$*:*

$$\left| \Pr\left[b = b'\right] - \frac{1}{2} \right| = \mathrm{negl}(\lambda).$$

We now show that any scheme that is passive-adaptively secure is also active-adaptively secure. The intuition here is that the challenge ciphertext cannot depend on public keys for which the attacker knows the secret (since otherwise the scheme would be trivially broken), so the queries to the malicious-corrupt oracles have no effect on the distribution of the challenge ciphertext.

Lemma 4.6 (Passive-Adaptive to Active-Adaptive)**.** *Let* Π_{dRBE} *be a passive-adaptively secure distributed broadcast encryption scheme. Then* Π_{dRBE} *is also active-adaptively secure.*

Proof Sketch. The reduction only forwards the messages of the adversary to the challenger, except for the queries to the malicious corrupt oracle, which the reduction simply discards. Since the challenge ciphertext is computed as

$$\mathsf{ct}^* \leftarrow \mathsf{Enc}(\mathsf{pp}, \{\mathsf{upk}_j\}_{j \in S^*}, S^*, M_b^*).$$

and the set S^* does not contain any corrupted index, the distribution simulated by the reduction is identical to the view that the adversary is expecting. Thus any advantage of the active-adaptive adversary immediately carries over to the (passive-adaptive) reduction. □

4.3 Logarithmic Updates

One drawback of the current syntax for distributed broadcast encryption is that the decrypter has to constantly check the public bulletin board for public keys, since every time a new user joins a new public key is added to the system. It is desirable to minimize the number of updates that each user must receive, without affecting the functionality. More precisely, in the current model, all users need to know $\{\mathsf{upk}_i : i \in [S]\}$ in order to decrypt a ciphertext encrypted with respect to S. In the following we show that we can get away with checking the bulletin board $\log L$ times, with a modest increase in ciphertext size. Before discussing how to turn a scheme into one with logarithmic updates, let us make the notion of update somewhat more formal: We now define the notion of U-updates and we say that a scheme has logarithmic updates if $U = O(\log L)$. The following definition implicitly assumes that users are indexed by their time of arrival.

Definition 4.7 (U-updates)**.** *A DBE scheme with parameter L has U-updates if, for all $i \in [L]$ there exists a series of subsets*

$$S_1 \subseteq \cdots \subseteq S_U \subseteq [L]$$

such that for any $T \in [L]$ and for any $S^* \subseteq [T]$ it holds that

$$\Pr\left[\mathsf{Dec}(\mathsf{pp}, \{\mathsf{upk}_j\}_{j\in S_i}, \mathsf{usk}_i, \mathsf{ct}, S^*, i) = M : \mathsf{Enc}(\mathsf{pp}, \{\mathsf{upk}_j\}_{j\in S^*}, S^*, M)\right] = 1.$$

where i is the largest index such that $|S_i| \leq T$.

The Transformation. In the following we describe a simple transformation that achieves exactly this. This transformation has been used many times in the literature [16, 22, 25, 12] and we report it here for completeness. For convenience, we are going to assume that the total number of users in the system is

$$L = 1 + 2 + 4 + \cdots + 2^\ell$$

which is without loss of generality, since we can always pad the number of users to that L is of this form. Let us describe the algorithms of the scheme.

- Setup: Sample the public parameter pp for a distributed broadcast encryption scheme with parameter L, and initialize $k = \ell + 1$ "master public keys" $\{\mathsf{mpk}_k = \emptyset\}_{k\in[\ell+1]}$.
- Key Generation: When the user indexed by j joins the system, it runs the key generation algorithm $\mathsf{KeyGen}(\mathsf{pp}, j)$ as prescribed by the scheme.
- Update: Upon each user joining the system, we will assign a k such that their public key is added to mpk_k. Let k be the smallest integer such that $\mathsf{mpk}_k = \emptyset$, the update algorithm proceeds as follows:
  - If $k = 1$, then we are simply going to assign $\mathsf{mpk}_1 = \{\mathsf{upk}_j\}$.
  - Else, we are going to set

    $$\mathsf{mpk}_k = \mathsf{mpk}_{k-1} \cup \cdots \cup \mathsf{mpk}_1 \cup \{\mathsf{upk}_j\}$$

    and reset $\mathsf{mpk}_{k-1} = \cdots = \mathsf{mpk}_1 = \emptyset$.

  Note that, by construction, the cardinality of mpk_k is either 0 or 2^{k-1}. Therefore, each user will only have to keep track of the master public key he currently resides into, along with its secret key.
- Encryption: Let S be the encryption subset and let us define $\{S_k = \emptyset\}_{k\in[\ell+1]}$. For each $j \in S$, based on the current number of users, we can derive the master public key where upk_j currently resides, which we denote by mpk_{k_j}. We then add $S_{k_j} \cup \{j\}$. The encrypter computes $\ell + 1$ ciphertexts as

  $$\left\{\mathsf{ct}_k \leftarrow \mathsf{Enc}(\mathsf{pp}, \{\mathsf{upk}_j\}_{j\in S_k}, S_k, M)\right\}_{k\in[\ell+1]}.$$

- Decryption: The j-th user decrypts the ciphertext corresponding to the mpk_k where their key currently resides.

It is important to observe that each user only has to keep track of the keys inside of her master public key (i.e., the bundle), so it will only have to update their keys whenever the public key is moved up to the next bundle. Since there are at most $\ell + 1 \approx \log(L)$ bundles, it follows that each user receives at most logarithmically many updates throughout the lifetime of the system.

As for the correctness of the scheme, it suffices to observe that, by construction $S = S_1 \cup \cdots \cup S_{\ell+1}$ and therefore correctness follow from the correctness of the underlying distributed broadcast encryption scheme. Security also follows from a standard hybrid argument.

5 Distributed Broadcast Encryption from decision Bilinear Diffie-Hellman Exponent

In this section we present our first Distributed Broadcast Encryption scheme from the decision Bilinear Diffie-Hellman Exponent assumption. For a more intuitive overview we refer to section 2.2.

5.1 Our Distributed Broadcast Encryption scheme

Below we describe our Distributed Broadcast Encryption scheme $\Pi_{\mathsf{DBE},1}$.

- $\mathsf{Setup}(1^\lambda, 1^L)$: On input the security parameter λ and the number of slots L, generate a bilinear group $\mathsf{bg} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, [1]_1, [1]_2, e) \leftarrow \mathcal{BG}(1^\lambda)$, sample $\alpha \leftarrow_\$ \mathbb{Z}_p^*$ and output the public parameters as:

$$\mathsf{pp} = \left(\mathsf{bg}, [\alpha]_1, \ldots, [\alpha^L]_1, [\alpha]_2, \ldots, [\alpha^L]_2, [\alpha^{L+2}]_2, \ldots, [\alpha^{2L}]_2\right)$$

  We denote $\mathsf{pp} := (\mathsf{pp}_0, \mathsf{pp}_1, \ldots, \mathsf{pp}_{3L})$ (defining $\mathsf{pp}_{2L+1} = [0]_2$).

- $\mathsf{KeyGen}(\mathsf{pp}, j)$: on input the public parameters $\mathsf{pp} := \left(\mathsf{bg}, \left\{[\alpha^j]_1\right\}_{j\in[L]}, \left\{[\alpha^j]_2\right\}_{j\in[2L],j\neq L+1}\right)$ and a slot j , sample the secret $\kappa_j \leftarrow_\$ \mathbb{Z}_p^*$ and output:

$$\begin{aligned}\mathsf{usk}_j &= [t_j\alpha^{L+1-j}]_2\\ \mathsf{upk}_j &= \left([t_j]_1, [t_j\alpha]_2, \ldots, [t_j\alpha^{L+1-j-1}]_2, [t_j\alpha^{L+1-j+1}]_2, \ldots, [t_j\alpha^L]_2\right)\end{aligned}$$

- $\mathsf{Enc}(\mathsf{pp}, \{\mathsf{upk}_j\}_{j\in S}, S, M)$: on input the public parameters $\mathsf{pp} := \left(\mathsf{bg}, \left\{[\alpha^j]_1\right\}_{j\in[L]}, \left\{[\alpha^j]_2\right\}_{j\in[2L],j\neq L+1}\right)$, a set $S \subseteq [L]$, the corresponding users' public keys $\mathsf{upk}_j := \left([t_j]_1, [t_j\alpha^\ell]_2\right)_{\ell\in[L],\ell\neq L+1-j}$ for each $j \in S$ and a message $M \in \mathbb{G}_T$, sample $s \leftarrow_\$ \mathbb{Z}_p^*$, compute $[s\alpha^{L+1}]_T = e\left([s\alpha]_1, [\alpha^L]_2\right)$ and output:

$$\mathsf{ct} = \left([s]_1, [s\sum_{j\in S}(t_j + \alpha^j)]_1, [s\alpha^{L+1}]_T \cdot M\right)$$

  where $[s\sum_{j\in S}(t_j + \alpha^j)]_1$ is computed as $\prod_{j\in S}([st_j]_1 \cdot [\alpha^j]_1)$ using upk_j and pp.

- $\mathsf{Dec}(\mathsf{pp}, \{\mathsf{upk}_j\}_{j\in S}, \mathsf{usk}_i, \mathsf{ct}, S, i) \to M$: on input the public parameters $\mathsf{pp} := \left(\mathsf{bg}, \left\{[\alpha^j]_1\right\}_{j\in[L]}, \left\{[\alpha^j]_2\right\}_{j\in[2L],j\neq L+1}\right)$, a set $S \subseteq [L]$, the corresponding users' public keys $\mathsf{upk}_j := \left([t_j]_1, [t_j\alpha^\ell]_2\right)_{\ell\in[L],\ell\neq L+1-j}$ for each $j \in S$, a slot index $i \in S$, the user's corresponding secret key $\mathsf{usk}_i := [t_i\alpha^{L+1-i}]_2$ and a ciphertext $\mathsf{ct} = (\mathsf{ct}_1, \mathsf{ct}_2, \mathsf{ct}_3)$ output:

$$M' = \mathsf{ct}_3 \cdot e\left(\mathsf{ct}_2^{-1}, \mathsf{pp}_{2L+1-i}\right) \cdot e\left(\mathsf{ct}_1, \mathsf{usk}_i \cdot \prod_{j\in S, j\neq i}\left(\mathsf{upk}_{j,L+1-i} \cdot \mathsf{pp}_{2L+1+j-i}\right)\right)$$

  where $\mathsf{pp}_{2L+1-i} := [-\alpha^{L+1-i}]_2$, $\mathsf{usk}_i := [t_i\alpha^{L+1-i}]_2$, $\mathsf{upk}_{j,L+1-i} := [t_j\alpha^{L+1-i}]_2$, $\mathsf{pp}_{2L+1+j-i} := [\alpha^{L+1+j-i}]_2$

Remark 5.1 (Storage requirements for the encryptor and decryptor)**.** *Syntactically speaking, in* DBE *to be able to encrypt and decrypt with respect to any arbitrary set $S \subseteq [L]$ one needs to store all public keys $\{\mathsf{upk}_j\}_{j\in[L]}$, of size $O(L^2)$ in the above scheme. However, concretely in our scheme it suffices to have linear size of information from $\{\mathsf{upk}_j\}_{j\in[L]}$:*

$$\begin{aligned}&\textit{Encryptor: } [t_1]_1, \ldots, [t_L]_1\\ &\textit{Decryptor } i\textit{: } [t_1\alpha^{L+1+1-i}]_2, \ldots, [t_{i-1}\alpha^{L}]_2, [t_{i+1}\alpha^{L+2}]_2, \ldots, [t_L\alpha^{L+1+L-i}]_2.\end{aligned}$$

To ease the presentation of the definition of DBE *we do not formalize this aspect of our concrete scheme, letting $\{\mathsf{upk}_j\}_{j\in S}$ to be the inputs of* Enc *and* Dec *in the formal description of the construction.*

5.2 Correctness

As noted in the definition of DBE correctness, in section 4.1, to prove correctness we first need to prove that there exists an isValid algorithm. We prove existence of such an algorithm for our scheme above by concretely constructing it.

Verifiable Keys. We define $\mathsf{isValid}(\mathsf{pp}, \mathsf{upk}^*_j, j)$ as follows:

- Parse $\mathsf{upk}^*_j := (\mathsf{upk}^*_{j,0}, \ldots, \mathsf{upk}^*_{j,L+1-j-1}, \mathsf{upk}^*_{j,L+1-j+1}, \ldots, \mathsf{upk}^*_{j,L})$.
- Then if the following holds output 1.

$$\begin{aligned}e\left(\mathsf{upk}^*_{j,0}, [\alpha^L]_2\right) = e\left([\alpha^{L-1}]_1, \mathsf{upk}^*_{j,1}\right) = \ldots = e\left([\alpha^j]_1, \mathsf{upk}^*_{j,L+1-j-1}\right) =\\ = e\left([\alpha^{j-2}]_1, \mathsf{upk}^*_{j,L+1-j+1}\right) = \ldots = e\left([1]_1, \mathsf{upk}^*_{j,L}\right)\end{aligned}$$

  otherwise output 0.

Since $\mathsf{upk}^*_{j,k} \in \mathbb{G}_2$ ($\mathsf{upk}^*_{j,0} \in \mathbb{G}_1$ resp.) and $\mathbb{G}_2$ ($\mathbb{G}_1$ resp.) has prime order, p, there exist $u_k \in \mathbb{Z}^*_p$ such that $\mathsf{upk}^*_{j,k} = [u_k]_2$ for all $k \in [L]$ ($\mathsf{upk}^*_{j,0} = [u_0]_1$ resp.). From this and the above paring checks we get that:

$$[u_0\alpha^L]_T = [u_1\alpha^{L-1}]_T = \ldots = [u_{j-1}\alpha^j]_T = [u_{j+1}\alpha^{j-2}]_T = \ldots = [u_L]_T$$

Which gives us that:

$$\mathsf{upk}^*_j = \left([u_0]_1, [u_0\alpha]_2, \ldots, [u_0\alpha^{L-j}]_2, [u_0\alpha^{L-j+2}]_2, \ldots, [u_0\alpha^L]_2\right).$$

Therefore if $\mathsf{isValid}(\mathsf{pp}, \mathsf{upk}^*_j, j) = 1$ then upk^*_j is in the support of KeyGen: $(\cdot, \mathsf{upk}^*_j) \leftarrow \mathsf{KeyGen}(\mathsf{pp}, j)$ for $t_j := u_0 \in \mathbb{Z}^*_p$ and $\mathsf{usk}_j = [u_0\alpha^{L+1-j}]_2$.

Correctness Let arbitrary $\lambda \in \mathbb{N}$, $L \in \mathbb{N}$, $\mathsf{pp} := \left(\mathsf{bg}, \left\{[\alpha^j]_1\right\}_{j\in[L]}, \left\{[\alpha^j]_2\right\}_{j\in[2L],j\neq L+1}\right)$ for some $\alpha \in \mathbb{Z}^*_p$, $\mathsf{usk}_i = [t_i\alpha^{L+1-i}]_2$, $\mathsf{upk}_i := \left([t_i]_1[t_i\alpha^k]_2\right)_{k\in[L],k\neq L+1-i}$ for some $t_i \in \mathbb{Z}^*_p$, upk_j such that $\mathsf{isValid}(\mathsf{pp}, \mathsf{upk}_j, j) = 1$ and for all $j \in [L]$. Furthermore, let $S \subseteq [L]$ such that $i \in S$ and $M \in \mathbb{M}$. Then $\mathsf{Enc}(\mathsf{pp}, \{\mathsf{upk}_j\}_{j\in S}, S, M)$ gives:

$$\mathsf{ct} = \left([s]_1, [s\sum_{j\in S}(t_j + \alpha^j)]_1, [s\alpha^{L+1}]_T \cdot M\right)$$

and $\mathsf{Dec}(\mathsf{pp}\{\mathsf{upk}_j\}_{j\in S}, \mathsf{usk}_i, \mathsf{ct}, S, i)$ gives:

$$
\begin{aligned}
M' &= \mathsf{ct}_3 \cdot e\left(\mathsf{ct}_2^{-1}, \mathsf{pp}_{2L+1-i}\right) \cdot e\left(\mathsf{ct}_1, \mathsf{usk}_i \cdot \prod_{j\in S, j\neq i} (\mathsf{upk}_{j,L+1-i} \cdot \mathsf{pp}_{2L+1+j-i})\right) \\
&= \mathsf{ct}_3 \cdot e\left(\mathsf{ct}_2, [-\alpha^{L+1-i}]_2\right) \cdot \\
&\qquad \cdot e\left(\mathsf{ct}_1, [t_i\alpha^{L+1-i}]_2 \cdot [\sum_{j\in S, j\neq i} (t_j\alpha^{L+1-i} + \alpha^{L+1+j-i})]_2\right) = \\
&= M \cdot [s\alpha^{L+1}]_T \cdot e\left([s\sum_{j\in S}(t_j + \alpha^j)]_1, [-\alpha^{L+1-i}]_2\right) \cdot \\
&\qquad \cdot e\left([s]_1, [t_i\alpha^{L+1-i} + \sum_{j\in S, j\neq i} (t_j\alpha^{L+1-i} + \alpha^{N+1+j-i})]_2\right) = \\
&= M \cdot [s\alpha^{L+1}]_T \cdot [s\sum_{j\in S}(t_j + \alpha^j)(-\alpha^{L+1-i})]_T \cdot \\
&\qquad \cdot [s(t_i\alpha^{L+1-i} + \sum_{j\in S, j\neq i} (t_j\alpha^{L+1-i} + \alpha^{N+1+j-i}))]_T = \\
&= M \cdot [s\alpha^{L+1} - s\sum_{j\in S}(t_j\alpha^{L+1-i} + \alpha^{L+1+j-i}) + st_i\alpha^{L+1-i} + \\
&\qquad + s\sum_{j\in S, j\neq i} (t_j\alpha^{L+1-i} + \alpha^{N+1+j-i})]_T = \\
&= M \cdot [s\alpha^{L+1} - s\alpha^{L+1}]_T = \\
&= M \cdot [1]_T = \\
&= M
\end{aligned}
$$

In the above equations we used the fact that since $\mathsf{isValid}(\mathsf{pp}, \mathsf{upk}_j, j) = 1$ the key upk_j is in the support of KeyGen (for some t_j).

5.3 Security

For the security of our scheme we rely on the (decisional) Bilinear Diffie Hellman Exponent assumption [4] (see section 3.1).

We prove our scheme semi-selectively secure (see section 4.2). Looking ahead, we can transform our semi-selective DBE scheme to a fully adaptive using the generic transformation of 4.2 (which extends the standard transformation of Gentry and Waters [21] to the distributed setting).

A more intuitive description of the security proof can be found in section 2.2. Below we formally state our main security theorem together with its proof.

Theorem 5.2. *If the decisional Bilinear Diffie-Hellman Exponent assumption holds, then* $\Pi_{\mathsf{DBE},1}$ *is a semi-selectively secure Distributed Broadcast Encryption scheme. More specifically for every PPT adversary* $\mathcal{A}$ *against the semi-selective security of the above DBE construction,* $\Pi_{\mathsf{DBE},1}$*, there exists a PPT adversary* $\mathcal{B}$

against the decisional Bilinear Diffie-Hellman Exponent assumption such that:

$$\mathbf{Adv}^{\Pi_{\mathsf{DBE},1}}_{L,\mathcal{A}}(\lambda) \leq \mathbf{Adv}^{\mathsf{dBDHE}}_{\mathcal{BG},L,\mathcal{B}}(\lambda).$$

Proof. Assume a PPT adversary $\mathcal{A}$ that wins the semi-selective security of the above Distributed Broadcast Encryption scheme with a non-negligible probability $\epsilon > 1/\operatorname{poly}(\lambda)$. Moreover let $\mathcal{B}$ be an adversary to the dBHE assumption. $\mathcal{B}$ plays the role of the challenger in the DBE semi-selective security game with $\mathcal{A}$ in order to win the game of the assumption (parametrized by $q = L$).

$\mathcal{B}$ takes as input bg, $\left\{[\alpha^j]_1\right\}_{j\in[L]}$, $\left\{[\alpha^j]_2\right\}_{j\in[2L],j\neq q+1}$, $[s]_1$ and T. The adversary $\mathcal{A}$ sends to $\mathcal{B}$ the target set S^* and then $\mathcal{B}$ responds with $\mathsf{pp} = \left(\mathsf{bg}, \left\{[\alpha^j]_1\right\}_{j\in[L]}, \left\{[\alpha^j]_2\right\}_{j\in[2L],j\neq L+1}\right)$, which is identically distributed to an honestly generated pp.

Key Generation Phase: Recall that in our security definition (Definition 4.1) we are only concerned about keys for $j \in S^*$, thus it is sufficient for $\mathcal{B}$ to simulate those keys. For each $j \in S^*$, $\mathcal{B}$ samples $\tilde{t}_j \leftarrow_\$ \mathbb{Z}_p^*$ and implicitly sets $t_j = \tilde{t}_j - \alpha^j$ (without knowing α). That is, using the assumption's inputs $[\alpha^j]_1$ and $\{[\alpha_j]_2\}_{j\in[2L],j\neq L+1}$, it computes:

$$\begin{aligned}
\mathsf{upk}_j &= \left([\tilde{t}_j - \alpha^j]_1, \{[\tilde{t}_j\alpha^\ell - \alpha^{\ell+j}]_2\}_{\ell\in[L],\ell\neq L+1-j}\right) := \\
&:= \left([\tilde{t}_j - \alpha^j]_1, \{[(\tilde{t}_j - \alpha^j)\alpha^\ell]_2\}_{\ell\in[L],\ell\neq L+1-j}\right) := \\
&:= \left([t_j]_1, \{[t_j\alpha^\ell]_2\}_{\ell\in[L],\ell\neq L+1-j}\right)
\end{aligned}$$

and sends upk_j to $\mathcal{A}$. Since $\tilde{t}_j$ is uniformly random so is $t_j := \tilde{t}_j - \alpha^j$, therefore upk_j is identically distributed to an honestly generated one.

Challenge phase: $\mathcal{A}$ sends $M_0^*, M_1^* \in \mathbb{G}_T$ and a new target set $S^{**} \subseteq S^*$. $\mathcal{B}$ samples a bit $b \leftarrow_\$ \{0,1\}$ and sets

$$\mathsf{ct}^* = \left([s]_1, [s\sum_{j\in S^{**}} \tilde{t}_j]_1, T \cdot M_b^*\right)$$

using the assumption's input and the previously sampled $\tilde{t}_j$'s. Note that since $S^{**} \subseteq S^*$ all the corresponding upk_j are simulated by $\mathcal{B}$. Thus $[s\sum_{j\in S^{**}} \tilde{t}_j]_1 := [s\sum_{j\in S^{**}}(t_j + \alpha^j)]_1$ which means that ct^* is identically distributed to an honestly generated ciphertext. Finally, $\mathcal{B}$ sends ct^* to $\mathcal{A}$.

At the end $\mathcal{A}$ sends her guess b'.

Note that if $T = [s\alpha^{L+1}]_T$ then the ciphertext ct^* is perfectly indistinguishable from a real one so $\Pr[b = b'] = \epsilon$. On the other hand if T is uniformly random then the ciphertext leaks nothing about M_b, so $\Pr[b = b'] = 1/2$. It follows directly that $\mathcal{B}$ has advantage ϵ in distinguishing between $T = [s\alpha^{L+1}]_T$ and a random T. In conclusion:

$$\mathbf{Adv}^{\mathsf{dBDHE}}_{\mathcal{BG},L,\mathcal{B}}(\lambda) = \epsilon$$

which contradicts the dBDHE assumption. □

After that, from lemmata 4.3 and 4.6 of section 4.2 we get a distributed broadcast encryption construction with adaptive security. We summarize this result in corollary 5.3 stated below:

Corollary 5.3. *Let $\Pi'_{\mathsf{DBE},1}$ be the Distributed Broadcast Encryption (*DBE*) scheme after applying the transformation of lemma 4.3 to the $\Pi_{\mathsf{DBE},1}$ DBE scheme described above. Then $\Pi'_{\mathsf{DBE},1}$ is an adaptively secure* DBE *scheme in the random oracle model, assuming that the decisional Bilinear Diffie-Hellman Exponent assumption holds.*

Remark 5.4. *As stated in Remark 4.4 we can instead achieve an adaptively secure* DBE *in the standard model, at the cost of an $|S|$-bit overhead on the size of the ciphertext.*

5.4 Efficiency

Here we describe the efficiency and the possible trade-offs of our scheme after the transformations of section 4.2 (Lemma 4.3) to achieve adaptive security. Also we assume a random oracle is used to output the S-sized bit-string and is given in the ciphertext.

The public parameters consist of $2L$ $\mathbb{G}_1$ and $4L$ $\mathbb{G}_2$-elements: $|\mathsf{pp}| = O(L)$. The secret key of a user is 2 $\mathbb{G}_2$-element: $|\mathsf{usk}_j| = O(1)$. The public key of each user is 2 $\mathbb{G}_1$ and $2(L-2)$ $\mathbb{G}_2$-elements: $|\mathsf{upk}_j| = O(L)$. The ciphertext size is 4 $\mathbb{G}_1$ and 2 $\mathbb{G}_T$-elements, independently of the size of the set of users S that encrypts the message for: $|\mathsf{ct}| = O(1)$. Finally both Enc and Dec run in time $O(|S|)$.

The overall information that has to be stored in the bulletin board has size $O(L^2)$, dominated by the L public keys $\{\mathsf{usk}_j\}_{j\in[L]}$ each of size $O(L)$. The storage overhead of an encryptor and any decryptor $i \in [L]$ is $O(L)$ (see remark 5.1).

Concretely: $|\mathsf{pp}| = 6L \cdot |\mathbb{G}_1|$, $|\mathsf{usk}_i| = 2 \cdot |\mathbb{G}_2|$, $|\mathsf{upk}_j| = 2 \cdot |\mathbb{G}_1| + (2L-2) \cdot |\mathbb{G}_2|$, $|\mathsf{ct}| = 4|\mathbb{G}_1| + 2|\mathbb{G}_T|$, $|\text{Encryptor}| = 2L|\mathbb{G}_1|$, $|\text{Decryptor}| = (2L-2)|\mathbb{G}_2|$, $|\text{BB}| = 2L|\mathbb{G}_1| + L(2L-2)|\mathbb{G}_2|$.

Logarithmic Updates. As discussed in section 4.3, without further care, each decryptor needs to update her view after each new public key is entering the system, which trivially results in $O(L)$ updates. In order to reduce the maximum necessary number of these updates to $O(\log L)$ we need to apply a standard transformation [16], which gives an $O(\log L)$ overhead to the size of the ciphertext.

Concretely: $|\mathsf{ct}| = 4 \log L|\mathbb{G}_1| + 2 \log L|\mathbb{G}_T|$, $|\text{Decryptor}| \in \{0, 2, 6, \ldots, 2 \cdot 2^k - 2, \ldots, 2L/2 - 2\}|\mathbb{G}_2|$ (worst case $(L-2)|\mathbb{G}_2|$) and the rest of the values remain as above.

Square-root trade-off. Another, standard in Broadcast Encryption, transformation [4, 18] that provides a trade-off between ciphertext size and storage requirements (i.e. pp and upk_j size) is to consider L^ϵ, $0 \leq \epsilon < 1$ number of sets of users. That is 'divide' $[L] = 1, \ldots, L$ in $B = L^\epsilon$ (assume wlog that B divides L) sets

$$A_1 = \{1, \ldots, \frac{L}{B}\}, A_2 = \{\frac{L}{B} + 1, \ldots, \frac{2L}{B}\}, \ldots, A_B = \{\frac{(B-1)L}{B} + 1, \ldots, L\}.$$

This results in $O(L^\epsilon)$-sized ciphertexts but smaller parameters and public keys, $O(L^{1-\epsilon})$. We note that, interestingly for the case of Distributed Broadcast Encryption, this additionally results in $O(L^{1-\epsilon})$ number of updates for the decryptor. One can also consider a combination of both the above transformations to reduce the number of decryptor's updates to $\log(L^{1-\epsilon})$ at the cost of increasing the ciphertext size to $O(L^\epsilon \log(L^{1-\epsilon}))$.

Concrete numbers. For the sake of concreteness, we consider an example use case where we have a relatively small number of users $L = 1024$ and the BLS12-381 curve instantiating the bilinear group. Then our (logarithmic updates) variant has:

- $|\mathsf{pp}| = 288$KB
- $|\mathsf{usk}_i| = 0.19$KB
- $|\mathsf{upk}_j| = 191.9$KB

- $|\text{BB}| = 191.9\text{MB}$
- $|\text{Encryptor}| = 96\text{KB}$
- $|\text{Decryptor}| = 95.1\text{KB}$ (worst case)
- $|\mathsf{ct}| = 13\text{KB}$
- Number of decryptor updates: 10

6 Distributed Broadcast Encryption from k-Lin

Now we present our second Distributed Broadcast Encryption scheme from a static assumption, k-Lin. We prove this scheme adaptively secure in the standard model using the standard "dual-system" methodology [32, 28]. For a more intuitive overview we refer to section 2.3.

6.1 Our Construction

Below we provide the description of our second Distributed Broadcast Encryption scheme $\Pi_{\mathsf{DBE},2}$.

- $\mathsf{Setup}(1^\lambda, 1^L)$: On input the security parameter λ and the number of slots L, generate a bilinear group $\mathsf{bg} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, [1]_1, [1]_2, e) \leftarrow \mathcal{BG}(1^\lambda)$, sample $\mathbf{A}^\top \leftarrow_\$ \mathcal{L}_k$, $\mathbf{k} \leftarrow_\$ \mathbb{Z}_p^{k+1}$ and $\mathbf{W} \leftarrow_\$ \mathbb{Z}_p^{(k+1)\times k}$, $\mathbf{r}_j \leftarrow_\$ \mathbb{Z}_p^k$ for each $j \in [L]$ and output the public parameters as:

$$\begin{aligned}\mathsf{pp} = \big(&\mathsf{bg}, [\mathbf{A}]_1, \{[\mathbf{A}\mathbf{W}_j]_1\}_{j\in[L]}, \{[\mathbf{r}_j]_2\}_{j\in[L]},\\ &\{[\mathbf{W}_\ell \mathbf{r}_j]_2\}_{\ell,j\in[L],\ell\neq j}, \{[\mathbf{k}+\mathbf{W}_j\mathbf{r}_j]_2\}_{j\in[L]}, [\mathbf{A}\mathbf{k}]_T\big)\end{aligned}$$

  We denote $\mathsf{pp} = \big(\mathsf{pp}^{(0)}, \mathsf{pp}^{(1)}, \mathsf{pp}^{(2)}, \mathsf{pp}^{(3)}, \mathsf{pp}^{(4)}, \mathsf{pp}^{(5)}, \mathsf{pp}^{(6)}\big)$

- $\mathsf{KeyGen}(\mathsf{pp}, j)$: on input the public parameters pp and a slot j , sample the secret $\mathbf{T}_j \leftarrow_\$ \mathbb{Z}_p^{(k+1)\times k}$ and output:

$$\begin{aligned}\mathsf{usk}_j &= [\mathbf{T}_j\mathbf{r}_j]_2\\ \mathsf{upk}_j &= ([\mathbf{A}\mathbf{T}_j]_1[\mathbf{T}_j\mathbf{r}_1]_2, \ldots, [\mathbf{T}_j\mathbf{r}_{j-1}]_2, [\mathbf{T}_j\mathbf{r}_{j+1}]_2, \ldots, [\mathbf{T}_j\mathbf{r}_L]_2)\end{aligned}$$

- $\mathsf{Enc}(\mathsf{pp}, \{\mathsf{upk}_j\}_{j\in S}, S, M)$: on input the public parameters pp, a set $S \subseteq [L]$, the corresponding users' public keys upk_j for each $j \in S$ and a message $M \in \mathbb{G}_T$, sample $\mathbf{s} \leftarrow_\$ \mathbb{Z}_p^k$ and output:

$$\mathsf{ct} = \left([\mathbf{s}^\top\mathbf{A}]_1, [\mathbf{s}^\top\mathbf{A}\sum_{j\in S}(\mathbf{T}_j + \mathbf{W}_j)]_1, [\mathbf{s}^\top\mathbf{A}\mathbf{k}]_T \cdot M\right)$$

  where $[\mathbf{s}\sum_{j\in S}(\mathbf{T}_j + \mathbf{A}_j)]_1$ is computed as $= \prod_{j\in S}([\mathbf{s}^\top\mathbf{A}\mathbf{T}_j]_1 \cdot [\mathbf{s}^\top\mathbf{A}\mathbf{W}_j]_1)$ using $\mathsf{upk}_{j,0}$ and pp.

- $\mathsf{Dec}(\mathsf{pp}, \{\mathsf{upk}_j\}_{j\in S}, \mathsf{usk}_i, \mathsf{ct}, S, i)$: on input the public parameters pp, a set $S \subseteq [L]$, the corresponding users' public keys upk_j for each $j \in S$, a slot index $i \in S$, the user's corresponding secret key $\mathsf{usk}_i := [\mathbf{T}_i\mathbf{r}_i]_2$ and a ciphertext $\mathsf{ct} = (\mathsf{ct}_1, \mathsf{ct}_2, \mathsf{ct}_3)$ output:

$$M' = \mathsf{ct}_3 \cdot e\left(\mathsf{ct}_1^{-1}, \mathsf{pp}_i^{(5)} \cdot \mathsf{usk}_i \cdot \prod_{j\in S, j\neq i}(\mathsf{upk}_{j,i} \cdot \mathsf{pp}_{j,i}^{(5)})\right) \cdot e\left(\mathsf{ct}_2, \mathsf{pp}_i^{(3)}\right)$$

where $\mathsf{pp}_i^{(5)} := [\mathbf{k} + \mathbf{W}_i\mathbf{r}_i]_2$, $\mathsf{usk}_i := [\mathbf{T}_i\mathbf{r}_i]_2$, $\mathsf{upk}_{j,i} := [\mathbf{T}_j\mathbf{r}_i]_2$, $\mathsf{pp}_{j,i}^{(5)} = [\mathbf{W}_j\mathbf{r}_i]$, $\mathsf{pp}_i^{(3)} := [\mathbf{r}_i]_2$

6.2 Correctness

Verifiable Keys We define $\mathsf{isValid}(\mathsf{pp}, \mathsf{upk}_j^*, j)$ as follows:

- Parse $\mathsf{upk}_j^* := (\mathsf{upk}_{j,0}^*, \ldots, \mathsf{upk}_{j,j-1}^*, \mathsf{upk}_{j,j+1}^*, \ldots, \mathsf{upk}_{j,L}^*)$.
- Then if the following holds output 1.

$$
\begin{aligned}
e\left(\mathsf{upk}_{j,0}^*, [\mathbf{r}_1]_2\right) &= e\left([\mathbf{A}]_1, \mathsf{upk}_{j,1}^*\right) \\
&\vdots \\
e\left(\mathsf{upk}_{j,0}^*, [\mathbf{r}_{j-1}]_2\right) &= e\left([\mathbf{A}]_1, \mathsf{upk}_{j,j-1}^*\right) \\
e\left(\mathsf{upk}_{j,0}^*, [\mathbf{r}_{j+1}]_2\right) &= e\left([\mathbf{A}]_1, \mathsf{upk}_{j,j+1}^*\right) \\
&\vdots \\
e\left(\mathsf{upk}_{j,0}^*, [\mathbf{r}_L]_2\right) &= e\left([\mathbf{A}]_1, \mathsf{upk}_{j,L}^*\right)
\end{aligned}
$$

otherwise output 0.

It is easy to see that of $(\cdot, \mathsf{upk}_j^*) \in \mathsf{KeyGen}(\mathsf{pp}, j)$ then $\mathsf{isValid}(\mathsf{pp}, \mathsf{upk}_j^*, j) = 1$.

On the other hand, if $\mathsf{isValid}(\mathsf{pp}, \mathsf{upk}_j^*, j) = 1$ then we get the following: Since $\mathsf{upk}_{j,k}^* \in \mathbb{G}_2^{k+1}$ ($\mathsf{upk}_{j,0}^* \in \mathbb{G}_1^{k\times k}$ resp.) and $\mathbb{G}_2$ ($\mathbb{G}_1$ resp.) has prime order, p, there exist $\mathbf{u}_{j,\ell} \in \mathbb{Z}_p^{k+1}$ such that $\mathsf{upk}_{j,\ell}^* = [\mathbf{u}_{j,\ell}]_2$ for all $\ell \in [L], \ell \neq j$ ($\mathsf{upk}_{j,0}^* = [\mathbf{U}_j]_1$ resp.). From this and the above paring checks we get that:

$$
\begin{aligned}
[\mathbf{U}_j\mathbf{r}_1]_T &= [\mathbf{A}\mathbf{u}_{j,1}]_T \\
&\vdots \\
[\mathbf{U}_j\mathbf{r}_{j-1}]_T &= [\mathbf{A}\mathbf{u}_{j,j-1}]_T \\
[\mathbf{U}_j\mathbf{r}_{j+1}]_T &= [\mathbf{A}\mathbf{u}_{j,j+1}]_T \\
&\vdots \\
[\mathbf{U}_j\mathbf{r}_L]_T &= [\mathbf{A}\mathbf{u}_{j,L}]_T
\end{aligned}
$$

Correctness Let arbitrary $\lambda \in \mathbb{N}$, $L \in \mathbb{N}$,

$$
\begin{aligned}
\mathsf{pp} = \big(&\mathsf{bg}, [\mathbf{A}]_1, \{[\mathbf{A}\mathbf{W}_j]_1\}_{j\in[L]}, \{[\mathbf{r}_j]_2\}_{j\in[L]}, \\
&\{[\mathbf{W}_\ell\mathbf{r}_j]_2\}_{\ell,j\in[L],\ell\neq j}, \{[\mathbf{k} + \mathbf{W}_j\mathbf{r}_j]_2\}_{j\in[L]}, [\mathbf{A}\mathbf{k}]_T\big)
\end{aligned}
$$

for some $\mathbf{A} \in \mathbb{Z}_p^{k\times(k+1)}$, $\mathbf{k} \in \mathbb{Z}_p^{k+1}$ and $\mathbf{W} \in \mathbb{Z}_p^{(k+1)\times k}$, $\mathbf{r}_j \in \mathbb{Z}_p^k$ for each $j \in [L]$, $\mathsf{usk}_i = [\mathbf{T}_j\mathbf{r}_j]_2$ and $\mathsf{upk}_i := ([\mathbf{A}\mathbf{T}_i]_1, [\mathbf{T}_i\mathbf{r}_\ell])_{\ell\in[L],\ell\neq i}$ for some $\mathbf{T}_i \in \mathbb{Z}_p^{(k+1)\times k}$. Moreover, let $\{\mathsf{upk}_j\}_{j\in[L],j\neq i}$ such that $\mathsf{isValid}(\mathsf{pp}, \mathsf{upk}_j, j) = 1$, $S \subseteq [L]$ such that $i \in S$ and $M \in \mathbb{M}$. Then $\mathsf{Enc}(\mathsf{pp}, \{\mathsf{upk}_j\}_{j\in S}, S, M)$ gives:

$$
\mathsf{ct} = \left([\mathbf{s}^\top\mathbf{A}]_1, [\sum_{j\in S}(\mathbf{s}^\top\mathbf{U}_j + \mathbf{s}^\top\mathbf{A}\mathbf{W}_j)]_1, [\mathbf{s}^\top\mathbf{A}\mathbf{k}]_T \cdot M\right)
$$

where $[\mathbf{U}_j]_1 = \mathsf{upk}_{j,0}$, and $\mathsf{Dec}(\mathsf{pp}\{\mathsf{upk}_j\}_{j\in S}, \mathsf{usk}_i, \mathsf{ct}, S, i)$ gives:

$$
\begin{aligned}
M' &= \mathsf{ct}_3 \cdot e\left(\mathsf{ct}_2, \mathsf{pp}_i^{(3)}\right) \cdot e\left(\mathsf{ct}_1^{-1}, \mathsf{pp}_i^{(5)} \cdot \mathsf{usk}_i \cdot \prod_{j\in S, j\neq i} (\mathsf{upk}_{j,i} \cdot \mathsf{pp}_{j,i}^{(5)})\right) = \\
&= M \cdot [\mathbf{s}^\top \mathbf{A}\mathbf{k}]_T \cdot e\left([\sum_{j\in S}(\mathbf{s}^\top \mathbf{U}_j + \mathbf{s}^\top \mathbf{A}\mathbf{W}_j)]_1, [\mathbf{r}_i]_2\right) \cdot \\
&\quad \cdot e\left([-\mathbf{s}^\top \mathbf{A}]_1, [\mathbf{k} + \mathbf{W}_i \mathbf{r}_i]_2 \cdot [\mathbf{T}_i \mathbf{r}_i]_2 \cdot \prod_{j\in S, j\neq i} ([\mathbf{u}_{j,i}]_2 \cdot [\mathbf{W}_j \mathbf{r}_i])\right) = \\
&= M \cdot [\mathbf{s}^\top \mathbf{A}\mathbf{k}]_T \cdot [\sum_{j\in S}(\mathbf{s}^\top \mathbf{A}\mathbf{u}_{j,i} + \mathbf{s}^\top \mathbf{A}\mathbf{W}_j \mathbf{r}_i)]_T \cdot \\
&\quad \cdot [-\mathbf{s}^\top \mathbf{A}\mathbf{k} - \mathbf{s}^\top \mathbf{A}\mathbf{W}_i \mathbf{r}_i - \mathbf{s}^\top \mathbf{A}\mathbf{T}_i \mathbf{r}_i - \sum_{j\in S, j\neq i} (\mathbf{s}^\top \mathbf{A}\mathbf{u}_{j,i} + -\mathbf{s}^\top \mathbf{A}\mathbf{W}_j \mathbf{r}_i)]_T = \\
&= M \cdot [\mathbf{s}^\top \mathbf{A}\mathbf{k}]_T \cdot [-\mathbf{s}^\top \mathbf{A}\mathbf{k}]_T = \\
&= M
\end{aligned}
$$

where in the above we used that $\mathbf{U}_j \mathbf{r}_i = \mathbf{A}\mathbf{u}_{j,i}$, since $\mathsf{isValid}(\mathsf{pp}, \mathsf{upk}_j, j) = 1$

6.3 Security

We base the security of our scheme to the k-Lin assumption [3, 24, 31], the statement of which is given in section 3.1.

We prove adaptive security, following the techniques of [19, Section 6]. Interestingly, we achieve adaptive security without the need of the [21] transformation, therefore our scheme is adaptively secure (while maintaining constant-sized ciphertexts) in the standard model.

Theorem 6.1. *If the k-Lin assumption holds, then $\Pi_{\mathsf{DBE},2}$ is an adaptively secure Distributed Broadcast Encryption scheme. More specifically for every PPT adversary $\mathcal{A}$ against the active-adaptive security of the above DBE construction, $\Pi_{\mathsf{DBE},2}$, there exists a PPT adversary $\mathcal{B}$ against the k-Lin assumption such that:*

$$\mathbf{Adv}^{\Pi_{\mathsf{DBE},2}}_{\mathcal{BG},L,\mathcal{A}}(\lambda) \leq (2L+1) \cdot \mathbf{Adv}^{k\text{-}Lin}_{\mathcal{BG},k,\mathcal{B}}(\lambda) + \frac{1}{p}.$$

Proof. We prove the theorem using a sequence of Hybrids: Game_0, $\mathsf{Game}_1, \ldots, \mathsf{Game}_L$. In Game_0 we switch to a "semi-functional" ciphertext. In Game_i $(1 \leq i \leq L)$ we switch to "semi-functional" keys $\mathsf{usk}_1, \ldots, \mathsf{usk}_i$ while maintaining the rest of the keys functional. In Game_L we have both the ciphertext and all the keys being semi-functional and it follows M_b is information-theoretically masked (unless with a negligible probability $1/p$).

Hybrid 0. The Game_0 is the same as the (adaptive) security game of definition 4.2 except we switch the ciphertext to "semi-functional". That is

$$\mathsf{ct} = \left([\mathbf{c}^\top]_1, [\mathbf{c}^\top \sum_{j\in S}(\mathbf{T}_j + \mathbf{W}_j)]_1, [\mathbf{c}^\top \mathbf{k}]_T \cdot M\right)$$

where $\mathbf{c} \leftarrow_{\$} \mathbb{Z}_p^{k+1}$.

It is straightforward to show that Game_0 is negligibly close to the original game: Then there is an PPT adversary $\mathcal{B}_0$ such that

$$|\mathbf{Adv}^{\Pi_{\mathsf{DBE},2}}_{\mathcal{BG},L,\mathcal{A}}(\lambda) - \mathbf{Adv}^{\mathsf{Game}_0}_{\mathcal{BG},L,\mathcal{A}}(\lambda)| \leq \mathbf{Adv}^{k\text{-Lin}}_{\mathcal{BG},L,\mathcal{B}_0}(\lambda).$$

Let $\mathcal{B}_0$ be an adversary to k-Lin, receiving $(\mathsf{bg}, [\mathbf{A}^\top]_1, [\mathbf{c}]_1)$, where $[\mathbf{c}]_1$ is either $[\mathbf{A}^\top \mathbf{s}]$ (for some random $\mathbf{s} \in \mathbb{Z}_p^k$) or random. $\mathcal{B}_0$ will use an adversary $\mathcal{A}'$ that can distinguish with probability $\epsilon > 1/\operatorname{poly}(\lambda)$ between Game_0 and the original game. $\mathcal{B}_0$ samples $\mathbf{W}_j, \mathbf{r}_j, \mathbf{k}, \mathbf{T}_j$ and computes the public parameters and all the keys of the system according to $\Pi_{\mathsf{DBE},2}$ honestly. Now if $\mathbf{c} = \mathbf{A}^\top \mathbf{s}$ then $\mathcal{B}_0$ perfectly simulates the original game otherwise if $\mathbf{c}$ is random perfectly simulates Game_0, which means that by answering the same bit as $\mathcal{A}'$, $\mathcal{B}_0$ gains advantage ϵ against the k-Lin game.

It will be useful for the next Hybrids to state that with probability $1 - 1/p$ over $\mathbf{c}$: $\mathbf{c}$ lies outside the row span of $\mathbf{A}$, so there exists a non-zero $\mathbf{a}^\perp \in \mathbb{Z}_p^{k+1}$ such that $\mathbf{A} \cdot \mathbf{a}^\perp = \mathbf{0}$ and $\mathbf{c} \cdot \mathbf{a}^\perp \neq 0$. Moreover, we can efficiently compute $\mathbf{a}^\perp$ given $\mathbf{A}, \mathbf{c}$.

Hybrid i Game_i is the same as Game_{i-1} except we replace $[\mathbf{k} + \mathbf{W}_i \mathbf{r}_i]_2$ with $[\mathbf{k} + \mathbf{W}_i \mathbf{r}_i + \delta_i \mathbf{a}^\perp]_2$, where $\delta_i \leftarrow \mathbb{Z}_p$. We claim that hybrids $i-1$ and i are computationally indistinguishable assuming k-Lin in $\mathbb{G}_2$, which tells us

$$\left(\mathbf{A}, \mathbf{A}\mathbf{W}_i, [\mathbf{W}_i \mathbf{r}_i]_2, [\mathbf{r}_i]_2, [\mathbf{W}_i \mathbf{B}]_2, [\mathbf{B}]_2\right) \approx_c \left(\mathbf{A}, \mathbf{A}\mathbf{W}_i, [\mathbf{W}_i \mathbf{r}_i + \delta_i \mathbf{a}^\perp]_2, [\mathbf{r}_i]_2, [\mathbf{W}_i \mathbf{B}]_2, [\mathbf{B}]_2\right)$$

where $\mathbf{B} \leftarrow \mathbb{Z}_p^{k \times k}$.[1] Then, as in the previous case, we build a reduction that reduces distinguishing between Hybrid $i-1$ and Hybrid i to k-Lin. In particular:

$$|\mathbf{Adv}^{\mathsf{Game}_{i-1}}_{\mathcal{BG},L,\mathcal{B}_{i-1}}(\lambda) - \mathbf{Adv}^{\mathsf{Game}_i}_{\mathcal{BG},L,\mathcal{B}_i}(\lambda)| \leq 2 \cdot \mathbf{Adv}^{k\text{-Lin}}_{\mathcal{BG},L,\mathcal{B}_0}(\lambda).$$

We proceed via a case analysis:

- $i \notin S$: the reduction samples $\mathbf{c}, \mathbf{k}, \mathbf{T}_1, \ldots, \mathbf{T}_L$ at random, and for all $j \neq i$: samples $\mathbf{W}_j, \tilde{\mathbf{r}}_j \leftarrow \mathbb{Z}_p^k$ and implicitly sets $\mathbf{r}_j := \mathbf{B}\tilde{\mathbf{r}}_j$. In particular, it can
  1. simulate the challenge ciphertext without knowing $\mathbf{W}_i$ since $i \notin S$;
  2. compute $[\mathbf{r}_j]_2, [\mathbf{T}_i \mathbf{r}_j]_2$ for all i, j using $[\mathbf{B}]_2, \tilde{\mathbf{r}}_j, \mathbf{T}_i$
  3. compute $[\mathbf{W}_j \mathbf{r}_i]_2, j \neq i$ using $[\mathbf{r}_i]_2, \mathbf{W}_j$
  4. compute $[\mathbf{W}_i \mathbf{r}_j]_2, j \neq i$ using $[\mathbf{W}_i \mathbf{B}], \tilde{\mathbf{r}}_j$
  5. compute $\mathsf{pp}, \mathsf{upk}_1, \ldots, \mathsf{upk}_L, \mathsf{usk}_1, \ldots, \mathsf{usk}_L$.
- $i \in S$: the reduction proceeds as in the previous case, with the following changes:
  1. samples $\tilde{\mathbf{T}}_i \leftarrow \mathbb{Z}_p^{(k+1) \times k}$ (instead of $\mathbf{T}_i$ at random) and implicitly sets $\mathbf{T}_i := \tilde{\mathbf{T}}_i - \mathbf{W}_i$;
  2. simulates $\mathbf{c}(\mathbf{T}_i + \mathbf{W}_i)$ in the challenge ciphertext using $\mathbf{c}\tilde{\mathbf{T}}_i$;
  3. simulates $[\mathbf{T}_i \mathbf{r}_j]_2 = [\tilde{\mathbf{T}}_i \mathbf{B}\tilde{\mathbf{r}}_j - \mathbf{W}_i \mathbf{B}\tilde{\mathbf{r}}_j]_2, j \neq i$ using $[\mathbf{W}_i \mathbf{B}]_2, [\mathbf{B}]_2$ along with $\tilde{\mathbf{T}}_i, \tilde{\mathbf{r}}_j$.

  In particular, we do not need to simulate $\mathsf{usk}_i = [\mathbf{T}_i \mathbf{r}_i]_2$ since the query is not allowed.

Formally, the reduction guesses at random which case we will fall into, and abort if the guess is wrong, incuring a factor 2 security loss. This step is essentially the same as [19, Lemma 6.3].

[1] k-Lin tells us that

$$[\mathbf{B}]_2, [\mathbf{r}_i]_2, [\mathbf{t}^\top \mathbf{B}]_2, [\mathbf{t}^\top \mathbf{r}_i] \approx_c [\mathbf{B}]_2, [\mathbf{r}_i]_2, [\mathbf{t}^\top \mathbf{B}]_2, [\mathbf{t}^\top \mathbf{r}_i + \delta_i]$$

Now, the reduction samples a random $\tilde{\mathbf{W}}_i$ and programs $\mathbf{W}_i = \tilde{\mathbf{W}}_i + \mathbf{a}^\perp \mathbf{t}$.

Hybrid 3. We complete the proof using an information-theoretic argument, which states that $\mathbf{ck}$ is uniformly random (and in turn perfectly masks M_b) given

$$\mathbf{A}, \mathbf{Ak}, \mathbf{c}, \mathbf{k} + \delta_i \mathbf{a}^{\perp}, i = 1, \ldots, L$$

This comes directly from [19, Lemma 6.4]. □

6.4 Efficiency

We compute the efficiency of our scheme for $k = 1$. As can be seen by the construction's description, the public parameters of our scheme are dominated by $O(L^2)$ vectors in $(\mathbb{G}_2)^2$, upk_j consists of $O(L)$ elements (1 in $\mathbb{G}_1$ and $L-1$ in $(\mathbb{G}_2)^2$) and usk_j is one vector of groups elements in $(\mathbb{G}_2)^2$. The cipertext has size $O(1)$ independently of the size of the set S. The remark 5.1 of the BDHE construction applies here as well giving $O(L)$-size storage for both the Encryptor and the Decryptor. Finally the overall information that has to be stored in the bulletin board is $O(L^2)$, which is the L upk_j's.

Concretely: $|\mathsf{pp}| = (2+L)|\mathbb{G}_1|+(2L^2+L)|\mathbb{G}_2|+1|\mathbb{G}_1|$, $|\mathsf{usk}_i| = 2|\mathbb{G}_2|$, $|\mathsf{upk}_j| = 1|\mathbb{G}_1|+(2L-2)|\mathbb{G}_2|$, $|\mathsf{ct}| = 3|\mathbb{G}_1| + 1|\mathbb{G}_T|$, $|\text{Encryptor}| = 2L|\mathbb{G}_1| + 1|\mathbb{G}_T|$, $|\text{Decryptor}| = 1|\mathbb{G}_1| + (4L-1)|\mathbb{G}_2|$, $|\text{BB}| = 2L|\mathbb{G}_1| + L(2L-2)|\mathbb{G}_2|$.

Concrete numbers. We provide concrete sizes of the parameters of the scheme for $L = 1024$ number of users and the BLS12-381 curve. Then our (logarithmic updates) variant has:

- $|\mathsf{pp}| = 192.2$KB
- $|\mathsf{usk}_i| = 0.19$KB
- $|\mathsf{upk}_j| = 191.9$KB
- $|\text{BB}| = 191.9$MB
- $|\text{Encryptor}| = 96.6$KB
- $|\text{Decryptor}| = 384$KB (worst case)
- $|\mathsf{ct}| = 7$KB
- Number of decryptor updates: 10

Acknowledgments

We would like to thank Brent Waters and David Wu for helpful discussions on [25]. G.M. was partially funded by the German Federal Ministry of Education and Research (BMBF) in the course of the 6GEM research hub under grant number 16KISK038 and by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy - EXC 2092 CASA – 390781972. D.K. received funding from projects from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program under project PICOCRYPT (grant agreement No. 101001283) and from the Spanish Government under project PRODIGY (TED2021-132464B-I00) funded by MCIN/AEI/10.13039/501100011033/ and the European Union NextGenerationEU/ PRTR.

References

[1] Shweta Agrawal, Daniel Wichs, and Shota Yamada. Optimal broadcast encryption from LWE and pairings in the standard model. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part I*, volume 12550 of *LNCS*, pages 149–178. Springer, Heidelberg, November 2020.

[2] Shweta Agrawal and Shota Yamada. Optimal broadcast encryption from pairings and LWE. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part I*, volume 12105 of *LNCS*, pages 13–43. Springer, Heidelberg, May 2020.

[3] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In Matthew Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 41–55. Springer, Heidelberg, August 2004.

[4] Dan Boneh, Craig Gentry, and Brent Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In Victor Shoup, editor, *CRYPTO 2005*, volume 3621 of *LNCS*, pages 258–275. Springer, Heidelberg, August 2005.

[5] Dan Boneh and Brent Waters. A fully collusion resistant broadcast, trace, and revoke system. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM CCS 2006*, pages 211–220. ACM Press, October / November 2006.

[6] Dan Boneh, Brent Waters, and Mark Zhandry. Low overhead broadcast encryption from multilinear maps. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 206–223. Springer, Heidelberg, August 2014.

[7] Dan Boneh and Mark Zhandry. Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 480–499. Springer, Heidelberg, August 2014.

[8] Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Candidate iO from homomorphic encryption schemes. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part I*, volume 12105 of *LNCS*, pages 79–109. Springer, Heidelberg, May 2020.

[9] Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Factoring and pairings are not necessary for IO: circular-secure LWE suffices. In Mikolaj Bojanczyk, Emanuela Merelli, and David P. Woodruff, editors, *49th International Colloquium on Automata, Languages, and Programming, ICALP 2022, July 4-8, 2022, Paris, France*, volume 229 of *LIPIcs*, pages 28:1–28:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[10] Zvika Brakerski and Vinod Vaikuntanathan. Lattice-inspired broadcast encryption and succinct ciphertext-policy abe. In *13th Innovations in Theoretical Computer Science Conference (ITCS 2022)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2022.

[11] Jie Chen, Romain Gay, and Hoeteck Wee. Improved dual system ABE in prime-order groups via predicate encodings. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 595–624. Springer, Heidelberg, April 2015.

[12] Nico Döttling, Dimitris Kolonelos, Russell W. F. Lai, Chuanwei Lin, Giulio Malavolta, and Ahmadreza Rahimi. Efficient laconic cryptography from learning with errors. In *EUROCRYPT 2023, Part III*, LNCS, pages 417–446. Springer, Heidelberg, June 2023.

[13] Alex Escala, Gottfried Herold, Eike Kiltz, Carla Ràfols, and Jorge Villar. An algebraic framework for Diffie-Hellman assumptions. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 129–147. Springer, Heidelberg, August 2013.

[14] Amos Fiat and Moni Naor. Broadcast encryption. In Douglas R. Stinson, editor, *CRYPTO'93*, volume 773 of *LNCS*, pages 480–491. Springer, Heidelberg, August 1994.

[15] Danilo Francati, Daniele Friolo, Monosij Maitra, Giulio Malavolta, Ahmadreza Rahimi, and Daniele Venturi. Registered (inner-product) functional encryption. *IACR Cryptol. ePrint Arch.*, page 395, 2023.

[16] Sanjam Garg, Mohammad Hajiabadi, Mohammad Mahmoody, and Ahmadreza Rahimi. Registration-based encryption: Removing private-key generator from IBE. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018, Part I*, volume 11239 of *LNCS*, pages 689–718. Springer, Heidelberg, November 2018.

[17] Sanjam Garg, Mohammad Hajiabadi, Mohammad Mahmoody, Ahmadreza Rahimi, and Sruthi Sekar. Registration-based encryption from standard assumptions. In Dongdai Lin and Kazue Sako, editors, *PKC 2019, Part II*, volume 11443 of *LNCS*, pages 63–93. Springer, Heidelberg, April 2019.

[18] Romain Gay, Iordanis Kerenidis, and Hoeteck Wee. Communication complexity of conditional disclosure of secrets and attribute-based encryption. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 485–502. Springer, Heidelberg, August 2015.

[19] Romain Gay, Lucas Kowalczyk, and Hoeteck Wee. Tight adaptively secure broadcast encryption with short ciphertexts and keys. In Dario Catalano and Roberto De Prisco, editors, *SCN 18*, volume 11035 of *LNCS*, pages 123–139. Springer, Heidelberg, September 2018.

[20] Romain Gay and Rafael Pass. Indistinguishability obfuscation from circular security. pages 736–749. ACM Press, 2021.

[21] Craig Gentry and Brent Waters. Adaptive security in broadcast encryption systems (with short ciphertexts). In Antoine Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 171–188. Springer, Heidelberg, April 2009.

[22] Noemi Glaeser, Dimitris Kolonelos, Giulio Malavolta, and Ahmadreza Rahimi. Efficient registration-based encryption. *Cryptology ePrint Archive*, 2022.

[23] Jens Groth, Markulf Kohlweiss, Mary Maller, Sarah Meiklejohn, and Ian Miers. Updatable and universal common reference strings with applications to zk-SNARKs. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part III*, volume 10993 of *LNCS*, pages 698–728. Springer, Heidelberg, August 2018.

[24] Dennis Hofheinz and Eike Kiltz. Secure hybrid encryption from weakened key encapsulation. In Alfred Menezes, editor, *CRYPTO 2007*, volume 4622 of *LNCS*, pages 553–571. Springer, Heidelberg, August 2007.

[25] Susan Hohenberger, George Lu, Brent Waters, and David J. Wu. Registered attribute-based encryption. In *EUROCRYPT 2023, Part III*, LNCS, pages 511–542. Springer, Heidelberg, June 2023.

[26] Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from well-founded assumptions. pages 60–73. ACM Press, 2021.

[27] Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from LPN over $\mathbb{F}_p$, DLIN, and PRGs in NC^0. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part I*, volume 13275 of *LNCS*, pages 670–699. Springer, Heidelberg, May / June 2022.

[28] Allison B. Lewko and Brent Waters. New techniques for dual system encryption and fully secure HIBE with short ciphertexts. In Daniele Micciancio, editor, *TCC 2010*, volume 5978 of *LNCS*, pages 455–479. Springer, Heidelberg, February 2010.

[29] Mary Maller, Sean Bowe, Markulf Kohlweiss, and Sarah Meiklejohn. Sonic: Zero-knowledge SNARKs from linear-size universal and updatable structured reference strings. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019*, pages 2111–2128. ACM Press, November 2019.

[30] Duong Hieu Phan, David Pointcheval, and Mario Strefler. Decentralized dynamic broadcast encryption. In Ivan Visconti and Roberto De Prisco, editors, *SCN 12*, volume 7485 of *LNCS*, pages 166–183. Springer, Heidelberg, September 2012.

[31] Hovav Shacham. A cramer-shoup encryption scheme from the linear assumption and from progressively weaker linear variants. Cryptology ePrint Archive, Report 2007/074, 2007. https://eprint.iacr.org/2007/074.

[32] Brent Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 619–636. Springer, Heidelberg, August 2009.

[33] Hoeteck Wee. Broadcast encryption with size $N^{1/3}$ and more from k-lin. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part IV*, volume 12828 of *LNCS*, pages 155–178, Virtual Event, August 2021. Springer, Heidelberg.

[34] Hoeteck Wee. Optimal broadcast encryption and CP-ABE from evasive lattice assumptions. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part II*, volume 13276 of *LNCS*, pages 217–241. Springer, Heidelberg, May / June 2022.

[35] Hoeteck Wee and Daniel Wichs. Candidate obfuscation via oblivious LWE sampling. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part III*, volume 12698 of *LNCS*, pages 127–156. Springer, Heidelberg, October 2021.

[36] Qianhong Wu, Bo Qin, Lei Zhang, and Josep Domingo-Ferrer. Ad hoc broadcast encryption. *work*, 12(13):24. https://crises-deim.urv.cat/web/docs/publications/conferences/318.pdf.

[37] Qianhong Wu, Bo Qin, Lei Zhang, and Josep Domingo-Ferrer. Ad hoc broadcast encryption (poster presentation). In Ehab Al-Shaer, Angelos D. Keromytis, and Vitaly Shmatikov, editors, *ACM CCS 2010*, pages 741–743. ACM Press, October 2010.

A An attack on [36] and a proof of (a variant of) [37]

As we discuss in the introduction, WQZD presented a DBE scheme from Bilinear Pairings in [37] (proceedings version). They also provide a security statement that assumes the decisional BDHE assumption, however without a security proof and a they point out to the 'full version' of the paper. In the longer version of the paper [36] they present a different construction with an alleged security proof.

In the section we first show an attack on the scheme of the full version [36]. Then we provide a security proof for a light variant of the short (conference) version of the paper [37]. However the proof is only for selective security and not semi-selective (and therefore it cannot be proven adaptively secure using the transformation of [21]).

To avoid confusion, we want to highlight that this is feasible because the two constructions in [37] and [36] respectively are not the same.

A.1 The attack

The Distributed Broadcast Encryption scheme presented in [36] (long version) is as follows:

$$\mathsf{pp} = ([1], [h_1], \ldots, [h_L]), \qquad \{h_i \leftarrow_\$ \mathbb{Z}_p^*\}_i$$

$$\mathsf{usk}^{(j)} = \left([-r_j^{(j)}], [r_j^{(j)} h_1], \ldots, [x^{(j)} + r_j^{(j)} h_j], \ldots, [r_j^{(j)} h_L]\right) \qquad , \{x^{(j)}, r_i^{(j)} \leftarrow_\$ \mathbb{Z}_p^*\}_i$$

$$\mathsf{upk}^{(j)} = \left(\begin{array}{cccccc} [-r_1^{(j)}] & & \ldots & [0] & \ldots & [-r_L^{(j)}] \\ [x^{(j)}]_T & & & & & \\ \hline [x^{(j)} + r_1^{(j)} h_1] & & \ldots & [0] & \ldots & [r_L^{(j)} h_1] \\ [r_1^{(j)} h_2] & & \ldots & [0] & \ldots & [r_L^{(j)} h_2] \\ \vdots & & \ddots & \vdots & \ddots & \vdots \\ [r_1^{(j)} h_L] & & \ldots & [0] & \ldots & [x^{(j)} + r_L^{(j)} h_L] \end{array}\right)$$

$$\mathsf{ct} = \left([\gamma], [\gamma \sum_{j \in S} h_j], [\gamma \sum_{j \in S} x^{(j)}]_T \cdot M\right), \qquad \gamma \leftarrow_\$ \mathbb{Z}_p^*$$

Decryption for user $i \in S$ is based on:

$$e\left(\begin{bmatrix} \overbrace{\sum_{k \in S}(x^{(k)} + r_i^{(k)} h_i)}^{\tilde{x} + \tilde{r}_i h_i} + \sum_{j \in S, j \neq i} \overbrace{\sum_{k \in S}(r_i^{(k)} h_j)}^{\tilde{r}_i h_j} \end{bmatrix}, \overbrace{[\gamma]}^{\mathsf{ct}_1}\right) e\left(\begin{bmatrix} \overbrace{\sum_{k \in S} -r_i^{(k)}}^{-\tilde{r}_i} \end{bmatrix}, [\gamma \sum_{j \in S} h_j]\right) = \overbrace{[\gamma \sum_{j \in S} x^{(j)}]_T}^{\mathsf{ct}_3 / M}$$

where $\tilde{x} = \sum_{j \in S} x^{(j)}$, $\tilde{r} = \sum_{j \in S} r_i^{(j)}$.

Essentially the decryption by user $i \in S$ works as:

$$e\left(\mathsf{usk}_i^{(i)} \prod_{k \in S, k \neq i} \mathsf{upk}_{i,i}^{(k)} \prod_{j \in S, j \neq i} \mathsf{usk}_j^{(i)} \prod_{k \in S, k \neq i} \mathsf{upk}_{j,i}^{(k)}, \mathsf{ct}_1\right) e\left(\mathsf{usk}_0^{(i)} \prod_{k \in S, k \neq i} \mathsf{upk}_{0,i}^{(k)}, \mathsf{ct}_2\right) = \mathsf{ct}_3 / M$$

Attack. We show that *anyone* that has access to the bulletin board, i.e. the public keys of the users $\{\mathsf{upk}^{(j)}\}_{j\in[n]}$, can decrypt *any* ciphetext and recover the message. The attacker does not need to be a participant having a public/secret key pairs.

The core observation is that $[\gamma x^{(i)}]_T$ can be publicly obtained as:

$$
\begin{aligned}
[\gamma x^{(i)}]_T &= e\left([x^{(i)} + r^{(i)}_{\ell^*} h_{\ell^*} + \sum_{j\in S, j\neq \ell^*} r^{(i)}_{\ell^*} h^{(i)}_j], [\gamma]\right) e\left([-r^{(i)}_{\ell^*}], [\gamma \sum_{j\in S} h_j]\right) \\
&= e\left(\mathsf{upk}^{(i)}_{\ell^*,\ell^*} \cdot \prod_{j\in S, j\neq \ell^*} \mathsf{upk}^{(i)}_{j,\ell^*}, \mathsf{ct}_1\right) e\left(\mathsf{upk}^{(i)}_{0,\ell^*}, \mathsf{ct}_2\right)
\end{aligned}
$$

which is publicly computable for any $\ell^* \neq i$.

Therefore, applying the above method we can publicly compute all $\{[\gamma x^{(i)}]_T\}_{i\in S}$. Then by multiplying we get $\prod_{i\in S}[\gamma x^{(i)}]_T = \mathsf{ct}_3/M$, which directly gives us a way to derive the message: $M = \frac{\mathsf{ct}_3}{\prod_{i\in S}[\gamma x^{(i)}]_T}$.

A.2 Security proof of (a slight variant of) [37]

Below we show a variant of [37]. The index j^* is the largest index in the set S.

$$
\begin{aligned}
\mathsf{pp} &= ([1]_1, [r_1]_2, \ldots, [r_L]_2)\,, && \{r_i \leftarrow_\$ \mathbb{Z}_p^*\}_i \\
\mathsf{ct} &= \left([s]_1, [s(\sum_{j\in S} w_j + \sum_{j'\notin S} v^{(j^*)}_{j'})]_1, [s\sum_{j\in S}\alpha_j]_T \cdot M\right), && s \leftarrow_\$ \mathbb{Z}_p^* \\
\mathsf{upk}_i &= [\alpha_i]_T, [w_i]_1, \{[\alpha_i + w_i r_j]_2, j\neq i\}, \{[v^{(i)}_1]_1, \ldots, [v^{(i)}_L]_1, [v^{(i)}_{j'} r_j]_2, j'\neq j\} \\
\mathsf{usk}_i &= [\alpha_i + w_i r_i]_2
\end{aligned}
$$

The main difference with the [37] scheme is that we substitute $w^{(j^*)}_{j'}$ in the ciphertext with $v^{(j^*)}_{j'}$, where $v^{(\ell)}_k$ now sampled independently and is part of the public key of user ℓ.

Theorem A.1. *If the decisional Bilinear Diffie-Hellman Exponent assumption holds, then the DBE scheme presented above is a selectively secure Distributed Broadcast Encryption scheme.*

Proof. Assume a PPT adversary $\mathcal{A}$ that wins the selective security of the above Distributed Broadcast Encryption scheme with a non-negligible probability $\epsilon > 1/\operatorname{poly}(\lambda)$. Moreover let $\mathcal{B}$ be an adversary to the dBHE assumption. $\mathcal{B}$ plays the role of the challenger in the DBE semi-selective security game with $\mathcal{A}$ in order to win the game of the assumption (parametrized by $q = L$).

$\mathcal{B}$ takes as input bg, $\left\{[\alpha^j]_1\right\}_{j\in[L]}$, $\left\{[\alpha^j]_2\right\}_{j\in[2L], j\neq L+1}$, $[s]_1$ and T. The adversary $\mathcal{A}$ sends to $\mathcal{B}$ the target set S^*.

Recall that in our security definition (Definition 4.1) we are only concerned about keys for $j \in S^*$, thus it is sufficient for $\mathcal{B}$ to simulate those keys. Let i^* be the last index of S^* (the one that has the $v^{(i^*)}_{j'}$-terms in the ciphertext). $\mathcal{B}$ and implicitly sets:

- $\alpha_{i^*} = \tilde{\alpha}_0 \alpha^{L+1}$, where $\tilde{\alpha}_0 \leftarrow_\$ \mathbb{Z}_p$ and $\alpha_j = \tilde{\alpha}_j$, where $\tilde{\alpha}_j \leftarrow_\$ \mathbb{Z}_p^*$, for each $j \in S^* \setminus \{i^*\}$.
- $r_j = \tilde{\alpha}_0 \alpha^{L+1-j} + \tilde{r}_j$, where $\tilde{r}_j \leftarrow_\$ \mathbb{Z}_p^*$, for each $j \in [L]$.

- $v_{j'}^{(i^*)} = \alpha^{j'} + \tilde{v}_{j'}^{(i^*)}$, where $\tilde{v}_{j'}^{(i^*)} \leftarrow_\$ \mathbb{Z}_p^*$, for each $j' \notin S^*$ and $v_j^{(i^*)} = \tilde{\beta}_j$, where $\tilde{\beta}_j \leftarrow_\$ \mathbb{Z}_p^*$, for each $j \in S^*$
- $w_j = \alpha^j + \tilde{w}_j$, where $\tilde{w}_j \leftarrow_\$ \mathbb{Z}_p^*$, for each $j \in S^* \setminus \{i^*\}$.
- $\tilde{u} = \sum_{j \in S} w_j + \sum_{j' \notin S} v_{j'}^{(i^*)}$ or $w_{i^*} = \tilde{u} - \sum_{j \in S, j \neq i^*} w_j - \sum_{j' \notin S} v_{j'}^{(i^*)}$, where $\tilde{u} \leftarrow_\$ \mathbb{Z}_p^*$.

Public Parameters Explicitly $\mathcal{B}$ sets: $r_j = [\tilde{\alpha}_0 \alpha^{L+1-j}]_2$ for each $j \in [L]$

$\mathcal{B}$ sets $\mathsf{pp} = \left([1]_1, \{[r_j]_2\}_{j \in [L]}\right)$ and sends pp to the adversary $\mathcal{A}$.

Notice that all the above terms are efficiently computable and that the simulated pp has the same distribution as the one.

Key Generation Phase: Now, explicitly the public keys $i \in S$ are simulated as:

- user $i \in S \setminus \{i^*\}$: First sample $v_j^{(i)} \leftarrow_\$ \mathbb{Z}_p^*$ and sets all $[v_j^{(i)}]_1, [v_{j'}^{(i)} r_i]_2$ as in the construction.

  Then set $[w_i]_1 = [\alpha^i]_1 \cdot [\tilde{w}_i]_1$, $[\alpha_i]_T = [\tilde{\alpha}_i]_T$. And for each $j \neq i$:

$$\begin{aligned}[\alpha_i + w_i r_j]_2 &= [\tilde{\alpha}_i]_2 \cdot [(\alpha^i + \tilde{w}_i)(\tilde{a}_0 \alpha^{L+1-j} + \tilde{r}_j)]_2 \\ &:= [\tilde{\alpha}_i]_2 \cdot [\tilde{a}_0 \alpha^{L+1-j+i} + \tilde{r}_j \alpha^i + \tilde{a}_0 \tilde{w}_i \alpha^{L+1-j} + \tilde{w}_j \tilde{r}_j]_2\end{aligned}$$

- user i^*: First sets
  - $v_j^{(i^*)} = [\tilde{\beta}_j]_1$ for each $j \in S^*$
  - $v_{j'}^{(i^*)} = [\alpha^{j'} + \tilde{v}_{j'}^{(i^*)}]_1$ for each $j \notin S^*$.
  - $[v_j^{(i^*)} r_i]_2 = [\tilde{\beta}_j \tilde{\alpha}_0 \alpha^{L+1-i}]_2$ for each $i \in [L]$, $j \in S^*$, $i \neq j$.
  - $[v_{j'}^{(i^*)} r_i]_2 = [(\alpha^{j'} + \tilde{v}_{j'}^{(i^*)}) \tilde{\alpha}_0 \alpha^{L+1-i}]_2 = [\tilde{\alpha}_0 \alpha^{L+1-i+j'} + \tilde{v}_{j'}^{(i^*)} \tilde{\alpha}_0 \alpha^{L+1-i}]_2$ for each $i \in [L]$, $j' \notin S^*$, $i \neq j'$.

  And then

$$\begin{aligned}[w_{i^*}]_1 &= [\tilde{u} - \sum_{j \in S^*, j \neq i^*} w_j - \sum_{j' \notin S^*} v_{j'}^{(i^*)}]_1 \\ &= [\tilde{u} - \sum_{j \in S^*, j \neq i^*} (\alpha^j + \tilde{w}_j) - \sum_{j' \notin S^*} (\alpha^{j'} + \tilde{v}_{j'}^{(i^*)})]_1 \\ [\alpha_i]_T &= [\tilde{\alpha}_i]_T\end{aligned}$$

  And for each $k \in S^* \setminus \{i^*\}$:

$$\begin{aligned}[\alpha_i + w_i r_k]_2 &= [\tilde{\alpha}_0 \alpha^{L+1} + \left(\tilde{u} - \sum_{j \in S^*, j \neq i^*} (\alpha^j + \tilde{w}_j) - \sum_{j' \notin S^*} (\alpha^{j'} + \tilde{v}_{j'}^{(i^*)})\right)(\tilde{a}_0 \alpha^{L+1-k} + \tilde{r}_k)]_2 \\ &:= [\tilde{\alpha}_0 \alpha^{L+1} + \tilde{u} \tilde{a}_0 \alpha^{L+1-k} - \sum_{j \in S^*, j \neq i^*} (\alpha^j + \tilde{w}_j) \tilde{a}_0 \alpha^{L+1-k} - \sum_{j' \notin S^*} (\alpha^{j'} + \tilde{v}_{j'}^{(i^*)}) \tilde{a}_0 \alpha^{L+1-k} \\ &\quad + \tilde{r}_k \tilde{u} - \tilde{r}_k \sum_{j \in S^*, j \neq i^*} (\alpha^j + \tilde{w}_j) - \tilde{r}_k \sum_{j' \notin S^*} (\alpha^{j'} + \tilde{v}_{j'}^{(i^*)})]_2 \\ &= [\tilde{u} \tilde{a}_0 \alpha^{L+1-k} - \sum_{j \in S^*, j \neq i^*, k} \tilde{a}_0 \alpha^{L+1-k+j} - \sum_{j \in S^*, j \neq i^*} \tilde{w}_j \tilde{a}_0 \alpha^{L+1-k} \\ &\quad - \sum_{j' \notin S^*} (\alpha^{L+1-k+j'} + \tilde{v}_{j'}^{(i^*)} \alpha^{L+1-k}) \tilde{a}_0 + \tilde{r}_k \tilde{u} - \tilde{r}_k \sum_{j \in S^*, j \neq i^*} (\alpha^j + \tilde{w}_j) - \tilde{r}_k \sum_{j' \notin S^*} (\alpha^{j'} + \tilde{v}_{j'}^{(i^*)})]_2\end{aligned}$$

Similarly for each $k \notin S^*$:

$$[\alpha_i + w_i r_k]_2 = [\tilde{u}\tilde{a}_0\alpha^{L+1-k} - \sum_{j\in S^*, j\neq i^*}(\alpha^{L+1-k+j} - \tilde{w}_j\alpha^{L+1-k})\tilde{a}_0 - \sum_{j'\notin S^*, j'\neq k}\alpha^{L+1-k+j'}\tilde{a}_0$$
$$- \sum_{j'\notin S^*}\tilde{v}_{j'}^{(i^*)}\alpha^{L+1-k}\tilde{a}_0 + \tilde{r}_k\tilde{u} - \tilde{r}_k\sum_{j\in S^*, j\neq i^*}(\alpha^j + \tilde{w}_j) - \tilde{r}_k\sum_{j'\notin S^*}(\alpha^{j'} + \tilde{v}_{j'}^{(i^*)})]_2$$

Notice that all terms are efficiently computable. Also all public keys are identically distributed to the real ones. $\mathcal{B}$then sends upk_j to $\mathcal{A}$.

Challenge phase: $\mathcal{A}$ sends $M_0^*, M_1^* \in \mathbb{G}_T$. $\mathcal{B}$ samples a bit $b \leftarrow_\$ \{0,1\}$ and sets

$$\mathsf{ct}^* = \left([s]_1, [s\tilde{u}]_1, T\cdot[s\sum_{j\in S^*, j\neq i^*}\tilde{\alpha}_j]_T \cdot M_b^*\right)$$

Finally, $\mathcal{B}$ sends ct^* to $\mathcal{A}$.

At the end $\mathcal{A}$ sends her guess b'.

Note that if $T = [s\alpha^{L+1}]_T$ then the ciphertext ct^* is perfectly indistinguishable from a real one so $\Pr[b = b'] = \epsilon$. On the other hand if T is uniformly random then the ciphertext leaks nothing about M_b, so $\Pr[b = b'] = 1/2$. It follows directly that $\mathcal{B}$ has advantage ϵ in distinguishing between $T = [s\alpha^{L+1}]_T$ and a random T. In conclusion:

$$\mathbf{Adv}^{\mathsf{dBDHE}}_{\mathcal{BG},L,\mathcal{B}}(\lambda) = \epsilon$$

which contradicts the dBDHE assumption. □