Fully Distributed Broadcast Encryption

Qianhong Wu[1,2], Bo Qin[1,3], Lei Zhang[4], and Josep Domingo-Ferrer[1]

[1] Universitat Rovira i Virgili, Dept. of Comp. Eng. and Maths
UNESCO Chair in Data Privacy, Tarragona, Catalonia
{qianhong.wu,bo.qin,josep.domingo}@urv.cat
[2] Key Lab. of Aerospace Information Security and Trusted Computing
Ministry of Education, School of Computer, Wuhan University, China
[3] Dept. of Maths, School of Science, Xi'an University of Technology, China
[4] Software Engineering Institute
East China Normal University, Shanghai, China
leizhang@sei.ecnu.edu.cn

Abstract. Broadcast encryption schemes rely on a centralized authority to generate decryption keys for each user. It is observed that, when a broadcast encryption scheme is deployed for secret escrows, a dishonest dealer can read the escrowed secrets without leaving any witnesses. We present a new broadcast encryption paradigm referred to as fully distributed broadcast encryption (FDBE) without suffering from this vulnerability. In the new paradigm, there are multiple dealers, and by contacting a number of them equal to a threshold or more, any user can join the system; then the secrets can be encrypted to any subset of users and only the intended receivers can decrypt, while an attacker cannot get any information about the encrypted message even if the attacker controls all the users outside the receiver set and corrupts some dealers, provided that the number of corrupted dealers is less than a threshold. We realize the first fully distributed broadcast encryption scheme which is proven secure under the decision Bilinear Diffie-Hellman Exponentiation assumption in the standard model. A variant is also shown to achieve sub-linear complexity in terms of public key, decryption key and ciphertext, comparable to up-to-date regular broadcast encryption schemes without robustness and strong security against misbehaving dealers.

Keywords: Broadcast encryption, Bilinear pairing, Provable security, Secrets escrow, Access control.

1 Introduction

Broadcasting is one of the most useful, versatile communication paradigms in distributed computing with many applications. Broadcast encryption (BE) [1] is a cryptographic primitive securing broadcast communication. In a setup phase, a dealer generates and privately distributes decryption keys to n users; then a sender can send a message to a dynamically chosen subset of receivers $\mathbb{R} \subseteq \{1, \cdots, n\}$ in such a way that only users in $\mathbb{R}$ can decrypt the ciphertext. Broadcast encryption schemes in the literatures can be classified into two categories:

symmetric key broadcast encryption and public key broadcast encryption. In the symmetric key setting, only the trusted center generates all the secret keys and broadcasts messages to users. Hence, only the key generation center can be the broadcaster. In the public key setting, in addition to the secret keys for each user, the trusted center also generates a public key for all the users so that any one can play the role of a broadcaster or sender.

Related Work. Fiat and Naor [1] first formalized broadcast encryption in the symmetric key setting and proposed a systematic method of broadcast encryption. The earlier broadcast encryption systems include a collusion bound: if an attacker compromises a number of users greater than the bound, the system no longer guarantees security even for encryptions solely sent to uncompromised users. Subsequent efforts were devoted to improve efficiency. Wallner *et al.* [2] and Wong [3] independently discovered the logical-tree-hierarchy scheme for group multicast. The parameters of the original schemes are improved in further work [4, 5]. Further improvements [7, 8, 9] reduced the decryption key size. Dodis and Fazio [10] extended the subtree difference method into a public-key broadcast system for a small size public key. D'Arco and Stinson's distributed BE scheme also follows this research line [11]. The state of the art is presented in [12] along this research line.

Public-key based BE schemes are more versatile in practice since it allows any one knowing the public key to broadcast safely. This avoids re-establishing the BE system for different applications. In the public key setting, Naor and Pinkas presented [13] the first public key broadcast encryption scheme. Their system is not collusion-resistant. If more than this threshold of users are revoked, then the scheme will be insecure and hence not fully collusion-resistant. Subsequently, by exploiting newly developed bilinear paring technologies, a fully collusion-resistant public key broadcast encryption was presented [14] which has $\mathcal{O}(\sqrt{n})$ complexity in key size, ciphertext size and computation cost, where n is the maximum allowable number of potential receivers. A recent scheme [15] reduces the key size and ciphertexts, although it has the same asymptotical complexity of sub-linear overhead as in [14]. The up-to-date schemes were presented in [16, 17, 18] which strengthen the security concept of public key BE schemes. However, as to system complexity, the sub-linear barrier $\mathcal{O}(\sqrt{n})$ has not yet been broken.

There are several extensions of the conventional BE notion. These variations can be easily mixed with our FDBE as they also make use of a threshold. But, unlike our motivation of removing the trust on the dealer in BE schemes, the efforts in the these variations have been devoted to achieving flexible decryption. These variants of BE systems were usually referred to as threshold broadcast encryption [19], dynamic threshold encryption [20, 21] or *ad hoc* threshold encryption [22, 23]. In such systems, a sender can choose a subset of receivers and a threshold, such that only a number of receivers equal to the threshold or more in the receiver set can cooperatively decrypt the ciphertext. An attacker controlling all the users outside the receiver set and some users in the receiver set cannot decrypt, provided that the number of corrupted users in the receiver set is less than the threshold. The state of the extensions was presented in [24] which achieves adaptive security and short ciphertexts.

Our Contribution. Existing BE systems rely on a fully trusted dealer that can reveal all the secrets encrypted by the senders without leaving any witnesses. This is not desirable in some applications. To relieve from the heavy dependance on the centralized dealer, we present a new notion referred to as fully distributed broadcast encryption (FDBE). In an FDBE system, each of multiple dealers has a public key and a combination (i.e., a simple multiplication operation in our instantiation) of each dealer's public key produces the system BE public key; by contacting a number of dealers equal to a threshold t or more, any user can obtain a decryption key; then any sender can send secret messages to any chosen users and only the intended users can decrypt. We define an attacker who can adaptively corrupt users and some dealers. Strong security is guaranteed if such a powerful attacker cannot decrypt the ciphertext, provided that the corrupted users are not in the receiver set and the number of corrupted dealers is less than the threshold. This implies that FDBE preserves security even if some dealers and users are corrupted, which is a desirable security property to remove the trust on the dealer and to preserve secrecy against compromised subscribers in BE systems. By exploiting pairings, we realize the first provably secure FDBE scheme under the decision Bilinear Diffie-Hellman Exponentiation (BDHE) assumption in the standard model (*i.e.* without using random oracles). The resulting scheme has sub-linear complexity in terms of public key, decryption key and ciphertext per user. This is comparable to up-to-date regular BE schemes which have no robustness or strong security against compromised dealers. Finally, we discuss the application of our FDBE notion to robust secrets escrow and controllable access to sensitive information.

2 Preliminaries

2.1 Bilinear Pairings and Assumptions

Our schemes are implemented in bilinear pairing groups [25]. Let `PairGen` be an algorithm that, on input a security parameter 1^λ, outputs a tuple $\Upsilon = (p, \mathbb{G}, \mathbb{G}_T, e)$, where $\mathbb{G}$ and $\mathbb{G}_T$ have the same prime order p, and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is an efficient non-degenerate bilinear map such that $e(g, g) \neq 1$ for any generator g of $\mathbb{G}$, and for all $u, v \in \mathbb{Z}$, it holds that $e(g^u, g^v) = e(g, g)^{uv}$.

The security of the schemes that we propose in this paper relies on the decision BDHE problem. The corresponding decision BDHE assumption is shown to be sound and used [18,26,27]. The decision BDHE assumption in $\mathbb{G}$ is as follows.

Definition 1 (Decision BDHE Assumption). *Let $\mathbb{G}$ and $\mathbb{G}_T$ be groups of order p with bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, and let g be a generator for $\mathbb{G}$. Let n be a nonnegative integer and let $\beta, \gamma \leftarrow \mathbb{Z}_p$ and $b \leftarrow \{0, 1\}$. If $b = 0$, set $Z = e(g, g)^{\beta^{n+1}\gamma}$; else, set $Z \leftarrow \mathbb{G}_T$. The problem instance consists of*

$$\{g^\gamma, Z\} \cup \{g^{\beta^i} : i \in [0, n] \cup [n + 2, 2n]\}$$

The Decision BDHE problem is to guess b. An attacker $\mathcal{A}$ wins if it correctly guesses b and its advantage is defined by $Adv_{BDHE_{\mathcal{A},n}}(\lambda) = |\Pr[\mathcal{A} \quad wins] - \frac{1}{2}|$. The Decision BDHE assumption states that, for any polynomial-time probabilistic attacker $\mathcal{A}$, $Adv_{BDHE_{\mathcal{A},n}}(\lambda)$ is negligible in λ.

2.2 Shamir's Secret Sharing Scheme

Our system exploits Shamir's (t, T)-threshold secret sharing scheme [28], which is briefly reviewed here. Let $\mathbb{Z}_p$ be a finite field with $p > T$ and $\xi \in \mathbb{Z}_p$ be the secret to be shared. A dealer picks a polynomial $f(\alpha)$ of degree at most $t-1$ at random, whose free term is the secret ξ, that is, $f(0) = \xi$. The polynomial $f(\alpha)$ can be written as $f(\alpha) = \xi + a_1\alpha + \cdots + a_{t-1}\alpha^{t-1} \mod p$, where $a_1, \cdots, a_{t-1} \in \mathbb{Z}_p$ are randomly chosen. Each shareholder k is assigned a known index $k \in \{1, \cdots, T\}$ and the dealer privately sends to shareholder k a share $\xi_k = f(k)$. Then any t holders in set $\mathbb{A} \subset \{1, \cdots, T\}$ can recover the secret $\xi = f(0)$ by interpolating their shares $\xi = f(0) = \sum_{k\in\mathbb{A}} \xi_k\lambda_k = \sum_{k\in\mathbb{A}} f(k)\lambda_k$, where $\lambda_k = \prod_{\ell\in\mathbb{A}}^{\ell\neq k} \frac{\ell}{\ell-k}$ are the Lagrange coefficients. Actually, shareholders in $\mathbb{A}$ can reconstruct the polynomial $f(\alpha) = \sum_{k\in\mathbb{A}} f(k)(\prod_{\ell\in\mathbb{A}}^{\ell\neq k} \frac{\ell-\alpha}{\ell-k})$.

3 Fully Distributed Broadcast Encryption

We first model FDBE systems and then formalize the security definitions in FDBE schemes. An attacker is allowed to adaptively corrupt some dealers and users before requesting the challenge ciphertext. This seems appropriate to capture powerful attackers against FDBE schemes.

3.1 Modeling FDBE Systems

For simplicity, we define FDBE as a key encapsulation mechanism. An FDBE system consists of the following polynomial-time algorithms:

- `Setup`(1^λ): This algorithm sets up the system. It takes as input a security parameter λ and outputs the global system parameters, including the number T of dealers, the threshold t, the number N of possible users, and the number n of the actual receivers of a broadcast. We denote the system parameters by π.
- `DKeyGen`(k, π): This key generation algorithm is run by each dealer $k \in \{1, \cdots, T\}$ to generate her public/private key pair. A dealer takes as input the system parameters π, her index $k \in \{1, \cdots, T\}$, and outputs $\langle X_k, x_k\rangle$ as her public/private key pair[1].

[1] In this definition, we distinguish the terms of private key, master secret key, decryption key share and decryption key. A private key is generated and held by a dealer. A master secret key is held by a dealer and it is jointly computed from the private keys of the dealers. A decryption key share is held by a user and it is computed from the private key of one dealer. A decryption key is held by a user and it is computed from that user's decryption key shares.

- `BEKeyGen`$(X_1, \cdots, X_T)$: This publicly computable algorithm generates the FDBE public encryption key. It takes as input all the dealers' public keys $X_1, \cdots, X_T$, and it outputs the FDBE public key PK.
- `MSKeyGen`$(x_1, \cdots, x_T)$: This is an interactive algorithm running among the dealers. It takes as input each dealer's private key, and it outputs S_k as dealer k's master secret key.
- `UKeyGen`(k, S_k, i): This algorithm is run by a dealer for index $k \in \{1, \cdots, T\}$ to generate a decryption key share for user $i \in \{1, \cdots, N\}$. For dealer k, the algorithm takes as input the dealer's index $k \in \{1, \cdots, T\}$ and master secret key S_k, and the user's index $i \in \{1, \cdots, N\}$, and it outputs $s_{i,k}$ as user i's decryption key share from dealer k. Denote user i's decryption key shares from dealers $k \in \mathbb{A} \subseteq \{1, \cdots, T\}$ by $s_{i,\mathbb{A}}$.
- `DKeyGen`$(i, s_{i,\mathbb{A}})$: This algorithm is run by each user to generate his decryption key. For user i, the algorithm takes as input the user's index $i \in \{1, \cdots, N\}$ and received decryption key shares $s_{i,\mathbb{A}}$, and it outputs d_i as the user's decryption key if $|\mathbb{A}| \geq t$.
- `Encryption`$(\mathbb{R}, PK)$: This algorithm is run by a sender to send messages to chosen users. It takes as input a recipient set $\mathbb{R} \subseteq \{1, \cdots, N\}$ and the FDBE public key PK. It outputs a pair $\langle Hdr, sk\rangle$, where Hdr is called the header and sk is the secret session key to encrypt messages. Let $\mathcal{E}_{sym}$ be a symmetric encryption scheme with key space $\mathbb{K}$, and $E(\cdot)$ and $D(\cdot)$ be the encryption and decryption algorithms, respectively. Let M be a message to be broadcast to the set $\mathbb{R}$, and let $C = E(M, sk)$ be the encryption of M under the session key $sk \in \mathbb{K}$. The information broadcast to users in $\mathbb{R}$ consists of $(\mathbb{R}, Hdr, C)$.
- `Decryption`$(\mathbb{R}, i, d_i, Hdr, PK)$: This algorithm allows each user in the receiver set to decrypt the message encryption key sk hidden in the header. It takes as input the receiver set $\mathbb{R}$, an index $i \in \{1, \cdots, N\}$, the receiver's secret key d_i, a header Hdr. If $i \in \mathbb{R}$, then the algorithm outputs the message encryption key sk. The key sk can then be used to decrypt C to obtain M by computing $M = D(C, sk)$.

3.2 Security Definitions

An FDBE scheme is correct if any user in the receiver set can decrypt a valid header, provided that the user has obtained sufficient decryption key shares from the dealers. A formal definition follows.

Definition 2 (Correctness). *An FDBE scheme is correct if for all* $\mathbb{R} \subseteq \{1, \cdots, N\}$*, all* $\mathbb{A} \subseteq \{1, \cdots, T\}$ $(|\mathbb{A}| \geq t)$*, all* $k \in \{1, \cdots, T\}$*,* $(X_k, x_k) \leftarrow$ *DKeyGen*(k, π)*,* $PK \leftarrow$ *BEKeyGen*$(X_1, \cdots, X_T)$*,* $S_k \leftarrow$ *MSKeyGen*$(x_1, \cdots, x_T)$*,* $s_{i,k} \leftarrow$ *UKeyGen*(k, S_k, i)*,* $d_i \leftarrow$ *DKeyGen*$(i, s_{i,\mathbb{A}})$*,* $\langle Hdr, sk\rangle \leftarrow$ *Encryption*$(\mathbb{R}, PK)$*, then Decryption*$(\mathbb{R}, i, d_i, Hdr, PK) = sk$ *for any* $i \in \mathbb{R}$*.*

We concentrate on adaptive security against corrupted dealers and users. For simplicity, we define security against chosen-plaintext attacks which are readily extended to capture chosen-ciphertext attacks. As usual, the attacker is allowed to see all the public data including the system parameters, each dealer's public key and the FDBE public key in an FDBE scheme. To capture *adaptive* security, the attacker is also allowed to adaptively ask for the private keys and master secret keys of several dealers, and the decryption key shares and the decryption keys of some users before choosing the set of indices that it wishes to attack. Formally, adaptive security in an MBDE scheme is defined using the following game between an attacker $\mathcal{A}$ and a challenger $\mathcal{CH}$. Both $\mathcal{CH}$ and $\mathcal{A}$ are given λ as input.

– `Setup.` The challenger runs $Setup(1^\lambda)$ to obtain the system parameters. The challenger gives the public system parameters to the attacker.
– `Corruption.` Attacker $\mathcal{A}$ can access all the public keys of the dealers and the FDBE public key. $\mathcal{A}$ can adaptively request the private keys and the master secret keys of dealers for some indices $i \in \{1, \cdots, T\}$, with a constraint that the number of the queried indices must be less than t. $\mathcal{A}$ can also adaptively request the decryption key shares and the decryption keys of users for some indices $i \in \{1, \cdots, N\}$.
– `Challenge.` At some point, $\mathcal{A}$ specifies a challenge set $\mathbb{R}^*$, such that for the decryption key of any user i queried in the corruption step, we have that $i \notin \mathbb{R}^*$ and, for any $i \in \mathbb{R}^*$, $\mathcal{A}$ has requested at most $t-1$ decryption key shares. The challenger sets $\langle Hdr^*, sk_0\rangle \leftarrow Encryption(\mathbb{R}^*, PK)$ and $sk_1 \leftarrow \mathbb{K}$. It sets $b \leftarrow \{0,1\}$ and gives (Hdr^*, sk_b) to $\mathcal{A}$.
– `Guess.` Attacker $\mathcal{A}$ outputs a guess bit $b' \in \{0,1\}$ for b and wins the game if $b = b'$.

We define $\mathcal{A}$'s advantage in attacking the FDBE system with security parameter λ as $Adv^{\mathtt{FDBE}}_{\mathcal{A},n,N,t,T}(1^\lambda) = |\Pr[b = b'] - \frac{1}{2}|$.

Definition 3 (Adaptive security). *We say that an FDBE scheme is adaptively secure if, for any polynomial-time algorithm $\mathcal{A}$, we have that $Adv^{FDBE}_{\mathcal{A},n,N,t,T}(1^\lambda)$ is negligible in λ.*

Similarly to conventional BE systems [18], another useful security definition is referred to as *semi-adaptive* security. In this game the attacker must commit to a set $\tilde{\mathbb{R}} \subseteq \mathbb{R}$ of indices at the *Initialization* phase. The attacker cannot query for the decryption key for any $i \in \tilde{\mathbb{R}}$. She can query for the decryption key shares for $i' \in \tilde{\mathbb{R}}$ but the queried decryption key shares must be less than t. She has to choose a target group $\tilde{\mathbb{R}}^*$ for the challenge ciphertext that is a subset of $\tilde{\mathbb{R}}$. A semi-adaptive attacker is weaker than an adaptive attacker, but stronger than a non-adaptive attacker since the attacker's choice of which subset of $\tilde{\mathbb{R}}$ to attack can be adaptive.

Remark 1. One may note that, although the attacker may be allowed to adaptively corrupt the dealers, their corruption is passive: a corrupted dealer will follow the protocol except that she will leak all her secret values and internal state information to the attacker. One may also define a stronger attacker by allowing a corrupted dealer to deviate from the protocol arbitrarily. This kind of attackers can be generally defeated by requiring the dealers to prove (in a zero-knowledge manner) that they have correctly followed the protocol in each step during the protocol execution. Hence, we concentrate on passive corruption in this paper.

3.3 Transforming Semi-adaptive Security to Adaptive Security

Recently, Gentry and Waters developed a modular proof approach [18] for adaptive security in regular BE systems with the two-key simulation technique [29]. We observe that this idea can also be employed to convert any semi-adaptively secure FDBE system into an adaptively secure system. The cost is doubling ciphertexts. Given a semi-adaptively secure FDBE system $\texttt{FDBE}_{SA}$ consists of algorithms: $Setup_{SA}$, $DKeyGen_{SA}$, $BEKeyGen_{SA}$, $MSKeyGen_{SA}$, $UKeyGen_{SA}$, $DKeyGen_{SA}$, $Encryption_{SA}$, $Decryption_{SA}$, we can build an adaptively secure $\texttt{FDBE}_{A}$ system as follows.

- $\texttt{Setup}(1^\lambda)$: Run $Setup_{SA}(1^\lambda)$ and obtain π including parameters $T, t, 2n, N$. This implies that if the underlying $\texttt{FDBE}_{SA}$ allows $2n$ users, then the adaptive variant allows n users.
- $\texttt{DKeyGen}(k, \pi)$: For $k = 1, \cdots, T$, Run $(X_k, x_k) \leftarrow DKeyGen_{SA}(k, \pi)$. Output (X_k, x_k) as dealer k's public/private key pair.
- $\texttt{BEKeyGen}(X_1, \cdots, X_T)$: Run $PK \leftarrow BEKeyGen_{SA}(X_1, \cdots, X_T)$. Output PK as the public broadcast encryption key.
- $\texttt{MSKeyGen}(x_1, \cdots, x_T)$: Run $(S'_1, \cdots, S'_T) \leftarrow MSKeyGen_{SA}(x_1, \cdots, x_T)$. Run a multiparty Mental Poker protocol [6] to generate random n-bit string $\theta \leftarrow \{0,1\}^n$. Set $S_k = (S'_k, \theta)(k = 1, \cdots, T)$. Output S_k as dealer k's master secret key. Denote the i-th bit of θ by θ_i.
- $\texttt{UKeyGen}(k, S_k, i)$: Run $s'_{i,k} \leftarrow UKeyGen_{SA}(k, S'_k, 2i - \theta_i)$. Set $s_{i,k} = (s'_{i,k}, \theta_i)$. Output $s_{i,k}$ as user i's decryption key share from dealer k. Denote $s'_{i,\mathbb{A}} = \{s'_{i,k} | k \in \mathbb{A}\}$.
- $\texttt{DKeyGen}(i, s_{i,\mathbb{A}})$: Run $d'_i \leftarrow DKeyGen_A(i, s'_{i,\mathbb{A}})$. Set $d_i = (d'_i, \theta_i)$. Output d_i as user i's decryption key.
- $\texttt{Encryption}(\mathbb{R}, PK)$: Generate a random set of $|\mathbb{R}|$ bits: $\zeta \leftarrow \{\zeta_i \leftarrow \{0,1\} : i \in \mathbb{R}\}$. Randomly choose $sk \leftarrow \mathbb{K}$. Set
  $$\mathbb{R}_0 \leftarrow \{2i - \zeta_i : i \in \mathbb{R}\}, \qquad \langle Hdr_0, sk_0 \rangle \leftarrow Encryption_{SA}(\mathbb{R}_0, PK);$$
  $$\mathbb{R}_1 \leftarrow \{2i - (1 - \zeta_i) : i \in \mathbb{R}\}, \langle Hdr_1, sk_1 \rangle \leftarrow Encryption_{SA}(\mathbb{R}_1, PK).$$
  Set
  $$C_0 \leftarrow E(M, sk_0), C_1 \leftarrow E(M, sk_1), Hdr \leftarrow \langle Hdr_0, C_0, Hdr_1, C_1, \zeta \rangle.$$
  Output $\langle Hdr, sk \rangle$.
- $\texttt{Decryption}(\mathbb{R}, i, d_i, Hdr, PK)$: Parse d_i as (d'_i, θ_i) and Hdr as $\langle Hdr_0, C_0, Hdr_1, C_1, \zeta \rangle$. Set $\mathbb{R}_0, \mathbb{R}_1$ as above. Run
  $$sk_{\theta_i \oplus \zeta_i} \leftarrow Decryption_{SA}(\mathbb{R}_{\theta_i \oplus \zeta_i}, i, d'_i, Hdr_{\theta_i \oplus \zeta_i}, PK), sk = D(C_{\theta_i \oplus \zeta_i}, sk_{\theta_i \oplus \zeta_i}).$$
  Output sk.

We briefly compare the Gentry-Waters conversion for regular BE systems with ours for FDBE systems. In the Gentry-Waters conversion for regular BE systems, each user is associated two potential secret decryption keys; however, the dealer gives the user only one of the two. An encryptor (who does not know which secret key the receiver possesses) encrypts the ciphertext twice, one for each key. The main benefit of this idea is that a simulator will have decryption keys for every user, and then it can always correctly answer the corruption queries from the attacker, hence circumventing the need of guessing of the target set in advance.

In our conversion, we employ a similar idea to associate one user with two potential decryption keys (corresponding to the FDBE public key) which can be reconstructed from the decryption key shares from the dealers. The difference is that, since there are multiple dealers, we allow multiple dealers to decide which decryption key a user can reconstruct. Then the encryptor and the users can do the same as in the Gentry-Waters conversion. It is easy to see that, from the perspective of a security proof, the two conversions are identical. This is due to the fact that, in the security proof of a regular BE system, a simulator will generate all the system parameters, the BE public key, the master secret key and the decryption keys on behalf of the dealer. In the context of FDBE systems, the simulator will do the same job on behalf of multiple dealers. In both cases, the attacker only communicates with the simulator. There is no difference for the attacker to communicate with the simulator in a regular BE or an FDBE system. Hence, the proof of the Gentry-Waters conversion can be trivially extended for the following theorem.

Theorem 1. *Let $\mathcal{A}$ be an adaptive attacker against $\mathtt{FDBE}_A$. Then, there exist algorithms $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3$, and $\mathcal{B}_4$, each running in about the same time as $\mathcal{A}$, such that*

$$Adv_{\mathcal{A},n,N,t,T}^{\mathtt{FDBE}_A}(\lambda) \leq Adv_{\mathcal{B}_1,2n,N,t,T}^{\mathtt{FDBE}_{SA}}(\lambda) + Adv_{\mathcal{B}_2,2n,N,t,T}^{\mathtt{FDBE}_{SA}}(\lambda) + Adv_{\mathcal{B}_3}^{\mathcal{E}_{sym}}(\lambda) + Adv_{\mathcal{B}_4}^{\mathcal{E}_{sym}}(\lambda),$$

where $Adv_{\mathcal{B}}^{\mathcal{E}_{sym}}(\lambda)$ is the advantage of an attacker $\mathcal{B}$ breaking the underlying symmetric encryption scheme.

Proof. It is omitted to avoid repetition.

4 Basic FDBE with Short Ciphertext

In this section, we propose a basic FDBE construction. The construction is built on the known Shamir secret sharing scheme [28] and the recent Gentry-Waters BE scheme [18]. The basic scheme has constant-size ciphertexts and is proven to be secure without using random oracles.

In the following proposed FDBE system, we assume that there are T dealers and n users. The T dealers jointly generate the public broadcast encryption key. By interacting with t dealers, for some threshold $t \leq T$, a user can obtain a secret decryption key. A sender can send confidential messages to any subset of the n users and only the chosen users can decrypt.

- **Setup.** Let `PairGen` be an algorithm that, on input a security parameter 1^λ, outputs a tuple $\Upsilon = (p, \mathbb{G}, \mathbb{G}_T, e)$, where $\mathbb{G}$ and $\mathbb{G}_T$ have the same prime order p, and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is an efficient non-degenerate bilinear map. Let $h_1, \cdots, h_n$ be randomly chosen from $\mathbb{G}$. The system parameters are $\pi = (\Upsilon, g, h_1, \cdots, h_n, t, T, n, N)$. For convenience, we denote $g = h_0$.
- **Broadcast key generation.** Each dealer k for $k = 1, \cdots, T$ generates a secret polynomial

$$f_k(\alpha) = x_k + a_{k,1}\alpha + \cdots + a_{k,t-1}\alpha^{t-1} \mod p,$$

where the randomly chosen $(x_k, a_{k,1}, \cdots, a_{k,t-1}) \in \mathbb{Z}_p^t$ is the private key of dealer k. Dealer k computes

$$X_k = e(g,g)^{x_k}, S_{k,\ell} = g^{f_k(\ell)} (\ell = 1, \cdots, T).$$

X_k is published as dealer k's public key. Dealer k privately sends $S_{k,\ell}$ to dealer ℓ. The public broadcast encryption key is

$$PK = \prod_{k=1}^{T} e(g,g)^{x_k} = e(g,g)^{x_1+\cdots+x_T} \overset{\mathsf{def}}{=} e(g,g)^x.$$

After receiving the secret key shares from other dealers, dealer k can extract her master secret key by computing $S_k = g^{f_1(k)} \cdots g^{f_T(k)} = g^{f_1(k)+\cdots+f_T(k)}$.
- **Decryption key distribution.** When receiving user i's request for a decryption key share $s_{i,k}$, the dealer k randomly chooses $r_{i,k} \in \mathbb{Z}_p$ and outputs:

$$s_{i,k} = \langle s_{i,0,k}, \cdots, s_{i,i-1,k}, s_{i,i,k}, s_{i,i+1,k}, \cdots, s_{i,n,k} \rangle$$

$$= \langle g^{-r_{i,k}}, h_1^{r_{i,k}}, \cdots, h_{i-1}^{r_{i,k}}, S_k h_i^{r_{i,k}}, h_{i+1}^{r_{i,k}}, \cdots, h_n^{r_{i,k}} \rangle.$$

Assume that user i has received decryption key shares $\{s_{i,k}\}_{k \in \mathbb{A}}$ from a set $\mathbb{A}$ of at least t dealers. The user i's decryption key is denoted by

$$d_i = \langle d_{i,0}, \cdots, d_{i,i-1}, d_{i,i}, d_{i,i+1}, \cdots, d_{i,n} \rangle.$$

We show each user can recover his/her decryption key with the received decryption key shares.Note that

$$g^{x_\ell} = g^{\sum_{k \in \mathbb{A}} f_\ell(k)\lambda_k}, \quad \text{and} \quad g^x = g^{\sum_{\ell=1}^{T} x_\ell} = g^{\sum_{\ell=1}^{T} \sum_{k \in \mathbb{A}} f_\ell(k)\lambda_k},$$

where λ_k is the Lagrange coefficient of Shamir's secret sharing scheme. Denote $r_i = \sum_{k \in \mathbb{A}} r_{i,k}\lambda_k$. It follows that

$$h_i^{r_j} = h_j^{\sum_{k \in \mathbb{A}} r_{i,k}\lambda_k} \quad \text{for} \quad j = 0, \cdots, n.$$

Hence, by respectively computing $d_{i,j} = \prod_{k \in \mathbb{A}} s_{i,j,k}^{\lambda_k}$, the received $s_{i,j,k}$'s in a multiplication way, the user i can reconstruct his/her decryption key[2]

$$\begin{aligned} d_i &= \langle d_{i,0}, \cdots, d_{i,i-1}, d_{i,i}, d_{i,i+1}, \cdots, d_{i,n} \rangle \\ &= \langle g^{-r_i}, h_1^{r_i}, \cdots, h_{i-1}^{r_i}, g^x h_i^{r_i}, h_{i+1}^{r_i}, \cdots, h_n^{r_i} \rangle. \end{aligned}$$

– **Broadcast encryption.** For a receiver set $\mathbb{R}$, randomly pick γ in $\mathbb{Z}_p$ and compute

$$Hdr = (c_1, c_2) : c_1 = g^\gamma, c_2 = (\prod_{j \in \mathbb{R}} h_j)^\gamma.$$

Set $sk = e(g,g)^{x\gamma}$ and output $\langle Hdr, sk \rangle$. Send $\langle \mathbb{R}, Hdr \rangle$ to receivers.

– **Broadcast decryption.** If $i \in \mathbb{R}$, user i can extract sk from Hdr with his decryption key d_i by computing

$$e(d_{i,i} \prod_{j \in \mathbb{R} \backslash \{i\}} d_{i,j}, c_1) e(d_{i,0}, c_2) = e(\prod_{j \in \mathbb{R}} d_{i,j}, c_1) e(d_{i,0}, c_2)$$

$$= e(g^x (\prod_{j \in \mathbb{R}} h_j)^{r_i}, g^\gamma)(g^{-r_i}, (\prod_{j \in \mathbb{R}} h_j)^\gamma) = e(g,g)^{x\gamma} = sk.$$

Theorem 2. *Let $\mathcal{A}$ be a semi-adaptive attacker breaking the above system with advantage ϵ in time τ. Then, there is an algorithm $\mathcal{B}$ breaking the BDHE assumption with advantage ϵ' in time τ', such that*

$$\epsilon' \geq \frac{1}{C_T^{t-1}} \epsilon, \tau' \leq \tau + \mathcal{O}(2\tau_{Pair} + (n^2 T^2 + T^4 + n^3)\tau_{Exp}),$$

where τ_{Pair} denotes the overhead to compute a pairing, and τ_{Exp} denotes the time complexity to compute one exponentiation without differentiating exponentiations in different groups.

We note that in Algorithm $\mathcal{B}$ there is an advantage loss by a factor $\frac{1}{C_T^{t-1}}$ w.r.t. Algorithm $\mathcal{A}$. However, since T and t are usually small, *e.g.*, $T \leq 10$, $t \leq 5$, the loss factor is $\frac{1}{C_T^{t-1}} \approx 10^{-3}$, which is not significant in practice.

5 Discussions

5.1 FDBE with Adaptive Security

The above construction only achieves semi-adaptive security. However, by applying the generic transformation from semi-adaptive security to fully-adaptive security in Section 3.3, the above scheme can be readily improved to meet fully-adaptive security, at the cost of doubling ciphertexts.

[2] If the user contacts a different set $\mathbb{A}'$ of at least t dealers, the user will recover another decryption key of the form $d_i' = \langle d_{i,0}', \cdots, d_{i,i-1}', d_{i,i}', d_{i,i+1}', \cdots, d_{i,n}' \rangle = \langle g^{-r_i'}, h_1^{r_i'}, \cdots, h_{i-1}^{r_i'}, g^x h_i^{r_i'}, h_{i+1}^{r_i'}, \cdots, h_n^{r_i'} \rangle$. They are equivalent for decryption. Furthermore, d_i' can be viewed as randomization of d_i and it can also be reconstructed by properly randomized shares from dealers in $\mathbb{A}$. Here, we say that a vector v' is a randomization of vector $v = (v_o, v_1, \cdots, v_n)$ by $\delta \in Z_p$ if $v' = (v_o g^\delta, v_1 h_1^\delta, \cdots, v_n h_n^\delta)$.

5.2 Tradeoff between Ciphertexts and Keys

In the above basic FDBE construction, the public broadcast key requires $\mathcal{O}(n)$ elements (by taking into account the system parameters), the decryption key of each user consists of $\mathcal{O}(n)$ elements and the ciphertext is $\mathcal{O}(1)$ size. In the following, we illustrate an efficient tradeoff between the decryption keys and ciphertexts.

Let $n_1 + \cdots + n_J = n$ and divide the maximal receiver group $\{i_1, \cdots, i_n\}$ into J subgroups each of which hosts at most n_j receivers in the j-th subgroup. Then when a sender wants to broadcast to a set of users $\mathbb{R} \subseteq \{i_1, \cdots, i_n\}$, the sender can concurrently apply our basic FDBE scheme to each subgroup $\mathbb{R}_j = \mathbb{R} \cap \{n_1 + \cdots + n_{j-1} + 1, \cdots, n_1 + \cdots + n_{j-1} + n_j\}$. After employing this approach, the public broadcast key, the decryption key of each user, and the ciphertext all consist of $\mathcal{O}(\sqrt{n})$ elements if $n_i = \cdots = n_J$ and $J = \sqrt{n}$.

This tradeoff approach is also applicable to the above adaptively secure variant. Hence, the resulting adaptively secure FDBE scheme has sub-linear complexity, *i.e.*, $\mathcal{O}(\sqrt{n})$ size public keys, decryption keys and ciphertexts. This performance is comparable to the up-to-date conventional broadcast schemes [14,16,17,18] which have also sub-linear complexity $\mathcal{O}(\sqrt{n})$.

5.3 Applications

The FDBE primitive has some promising applications. One of them is to enable robust and flexible access control of sensitive documents. For instance, unauthorized distribution of submissions to some conference has been an issue. Our FDBE can provide a solution to this problem. In first step, several co-chairs jointly initialize an FDBE system and distribute decryption keys to the committee members. In the second step, the authors provide the title and abstract of their submissions to the conference chairs. Then the chairs assign the review tasks to the committee members according to their bids and return the indices (i.e., one-time pseudonyms) of the reviewers to the authors. Finally, the authors encrypt their full submissions so that only assigned reviewers can read the full submissions. With this approach, unauthorized distribution will be greatly limited since any submission can only be accessed by the designated reviewers.

Another application is to support online pay-per-view movies against dishonest dealers using the D'Arco-Stinson framework [11]. Unlike the D'Arco-Stinson solution which employs symmetric-key BE scheme, our scheme is public key based and the system does not need to be re-established in case of premiere of a new movie. The movie producer only needs to know the indices of the new subscribers. Also, since our scheme is collusion resistant, the end users (together with at most $t - 1$ misbehaving dealers) cannot collude to decrypt and watch the movie if they are not subscribers of the corresponding movie.

6 Conclusion

To relieve from the dependence on the trustability of the single dealer in existing BE systems, we presented the new FDBE notion. In an FDBE system, multiple dealers jointly generate the BE public key; by contacting a number of dealers equal to a threshold or more, any user can obtain a secret decryption key; then a sender can send secret messages to any chosen users and only the intended users can decrypt. No attacker can decrypt, even if the attacker controls all the users outside the receiver set and corrupts some dealers, provided that the number of corrupted dealers is less than the threshold. We realized the first FDBE scheme proven to be secure under recognized assumptions. We also show how to implement our FDBE scheme for practical applications.

Acknowledgments. The authors are partly supported by by the EU 7FP through project "DwB", the Spanish Government through projects CTV-09-634, PTA2009-2738-E, TSI-020302-2010-153, PT-430000- 2010-31, TIN2009-11689, CONSOLIDER INGENIO 2010 "ARES" CSD2007-0004 and TSI2007-65406-C03-01, by the Government of Catalonia under grant SGR2009-1135, and by the Chinese NSF projects 60970115, 60970116, 61003214 and 61021004. The authors also acknowledge support by the Fundamental Research Funds for the Central Universities of China to Project 3103004, Beijing Municipal Natural Science Foundation to Project 4112052 and Shaanxi Provincial Education Department through Scientific Research Program 2010JK727. The fourth author is partially supported as an ICREA-Acadèmia researcher by the Catalan Government. The authors are with the UNESCO Chair in Data Privacy, but this paper does not necessarily reflect the position of UNESCO nor does it commit that organization.

References

1. Fiat, A., Naor, M.: Broadcast encryption. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 480–491. Springer, Heidelberg (1994)
2. Wallner, D.M., Harder, E.J., Agee, R.C.: Key Management for Multicast: Issues and Architectures. RFC Archives, #RFC2627 (1999)
3. Wong, C.K., Gouda, M., Lam, S.: Secure Group Communications Using Key Graphs. IEEE/ACM Trans. Netw. 8(1), 16–30 (2000)
4. Canetti, R., Garay, J., Itkis, G., Micciancio, D., Naor, M., Pinkas, B.: Multicast Security: A Taxonomy and Some Efficient Constructions. In: IEEE INFOCOM 1999, vol. 2, pp. 708–716. IEEE Press, New York (1999)
5. Canetti, R., Malkin, T., Nissim, K.: Efficient Communication-Storage Tradeoffs for Multicast Encryption. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 459–474. Springer, Heidelberg (1999)
6. Golle, P.: Dealing Cards in Poker Games. In: ITCC 2005, vol. 1, pp. 506–511. IEEE Press, Las Vegas (2005)
7. Halevy, D., Shamir, A.: The LSD Broadcast Encryption Scheme. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 47–60. Springer, Heidelberg (2002)
8. Goodrich, M.T., Sun, J.Z., Tamassia, R.: Efficient Tree-Based Revocation in Groups of Low-State Devices. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 511–527. Springer, Heidelberg (2004)
9. Sherman, A.T., McGrew, D.A.: Key Establishment in Large Dynamic Groups using One-way Function Trees. IEEE Trans. Softw. Eng. 29(5), 444–458 (2003)

10. Dodis, Y., Fazio, N.: Public Key Broadcast Encryption for Stateless Receivers. In: Feigenbaum, J. (ed.) DRM 2002. LNCS, vol. 2696, pp. 61–80. Springer, Heidelberg (2003)
11. D'Arco, P., Stinson, D.R.: Fault Tolerant and DistributedBroadcast Encryption. In: Joye, M. (ed.) CT-RSA 2003. LNCS, vol. 2612, pp. 263–280. Springer, Heidelberg (2003)
12. Cheon, J.H., Jho, N.S., Kim, M.H., Yoo, E.S.: Skipping, Cascade, and Combined Chain Schemes for Broadcast Encryption. IEEE Trans. Inf. Theory 54(11), 5155–5171 (2008)
13. Naor, M., Pinkas, B.: Efficient Trace and Revoke Schemes. In: Frankel, Y. (ed.) FC 2000. LNCS, vol. 1962, pp. 1–20. Springer, Heidelberg (2001)
14. Boneh, D., Gentry, C., Waters, B.: Collusion Resistant Broadcast Encryption with Short Ciphertexts and Private Keys. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 258–275. Springer, Heidelberg (2005)
15. Park, J.H., Kim, H.J., Sung, M.H., Lee, D.H.: Public Key Broadcast Encryption Schemes With Shorter Transmissions. IEEE Trans. on broadcasting 54(3), 401–411 (2008)
16. Boneh, D., Sahai, A., Waters, B.: Fully Collusion Resistant Traitor Tracing with Short Ciphertexts and Private Keys. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 573–592. Springer, Heidelberg (2006)
17. Boneh, D., Waters, B.: A Fully Collusion Resistant Broadcast, Trace, and Revoke System. In: Juels, A., Wright, R.-N., De Capitani di, V.S. (eds.) ACM CCS 2006, pp. 211–220. ACM Press, New York (2006)
18. Gentry, C., Waters, B.: Adaptive Security in Broadcast Encryption Systems (with Short Ciphertexts). In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 171–188. Springer, Heidelberg (2009)
19. Daza, V., Herranz, J., Morillo, P., Ràfols, C.: CCA2-Secure Threshold Broadcast Encryption with Shorter Ciphertexts. In: Susilo, W., Liu, J.K., Mu, Y. (eds.) ProvSec 2007. LNCS, vol. 4784, pp. 35–50. Springer, Heidelberg (2007)
20. Ghodosi, H., Pieprzyk, J., Safavi-Naini, R.: Dynamic Threshold Cryptosystems: A New Scheme in Group Oriented Cryptography. In: Pragocrypt 1996, pp. 370–379. CTU Publishing House (1996)
21. Lim, C.H., Lee, P.J.: Directed Signatures and Application to Threshold Cryptosystems. In: Lomas, M. (ed.) Security Protocols 1996. LNCS, vol. 1189, pp. 131–138. Springer, Heidelberg (1997)
22. Daza, V., Herranz, J., Morillo, P., Ràfols, C.: Ad-hoc Threshold Broadcast Encryption with Shorter Ciphertexts. Electronic Notes in Theoretical Computer Science 192(22), 3–5 (2008)
23. Delerablée, C., Pointcheval, D.: Dynamic Threshold Public-Key Encryption. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 317–334. Springer, Heidelberg (2008)
24. Qin, B., Wu, Q., Zhang, L., Domingo-Ferrer, J.: Threshold Public-Key Encryption with Adaptive Security and Short Ciphertexts. In: Soriano, M., Qing, S., López, J. (eds.) ICICS 2010. LNCS, vol. 6476, pp. 62–76. Springer, Heidelberg (2010)
25. Boneh, D., Franklin, M.: Identity Based Encryption from the Weil Pairing. SIAM J. of Computing 32(3), 586–615 (2003)
26. Boneh, D., Boyen, X., Goh, E.-J.: Hierarchical Identity Based Encryption with Constant Size Ciphertext. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 440–456. Springer, Heidelberg (2005)

27. Wu, Q., Mu, Y., Susilo, W., Qin, B., Domingo-Ferrer, J.: Asymmetric Group Key Agreement. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 153–170. Springer, Heidelberg (2009)
28. Shamir, A.: How to Share a Secret. Communications of the ACM 22, 612–613 (1979)
29. Katz, J., Wang, N.: Efficiency Improvements for Signature Schemes with Tight Security Reductions. In: Jajodia, S., Atluri, V., Jaeger, T. (eds.) ACM CCS 2003, pp. 155–164. ACM Press, New York (2003)
30. Gennaro, R., Jarecki, S., Krawczyk, H., Rabin, T.: Secure Distributed Key Generation for Discrete-Log Based Cryptosystems. J. Cryptology 20(1), 51–83 (2007)
31. Pedersen, T.P.: A Threshold Cryptosystem without a Trusted Party. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 522–526. Springer, Heidelberg (1991)

Appendix: Proof of Theorem 2

The proof outline is as follows. We construct an algorithm $\mathcal{B}$ to break an instance of the $BDHE$ assumption by invoking Attacker $\mathcal{A}$ against our scheme as a black box. $\mathcal{B}$ is given an instance of the $BDHE$ challenge. With it, $\mathcal{B}$ simulates the system parameters, the public keys of the dealers, private keys and secret key shares of corrupted dealers, and the decryption key shares of the users which the attacker may query for. The simulated data are indistinguishable from those generated in a real scheme from the viewpoint of the attacker, so that the attacker does not know she is interacting with a simulator. Then $\mathcal{B}$ uses $\mathcal{A}$'s guess to solve the $BDHE$ challenge. This contradicts the $BDHE$ assumption. Accordingly, such a successful attacker against our FDBE scheme does not exist and our scheme is secure.

Proof. $\mathcal{B}$ receives the BDHE challenge instance, which includes g^γ, Z and the set $\{g^{\beta^i} : i \in [0, n] \cup [n+2, 2n]\}$.

Init. $\mathcal{A}$ commits to a set $\bar{\mathbb{R}} \subseteq [1, n]$. $\mathcal{A}$ is allowed to obtain the system parameters, each dealer's public key (and the public broadcast encryption key), the private keys of the corrupted dealers, the private key shares of the corrupted dealers from both corrupted and non-corrupted dealers, the decryption key and the decryption key shares of the users in $[1, n] \setminus \bar{\mathbb{R}}$, and less than t decryption key shares of any user in $\bar{\mathbb{R}}$.

Setup. $\mathcal{B}$ generates $y_0, \cdots, y_n \leftarrow \mathbb{Z}_p$. It sets $h_i = g^{y_i}$ for $i \in \bar{\mathbb{R}}$, $h_i = g^{y_i + \beta^i}$ for $i \in [1, n] \setminus \bar{\mathbb{R}}$. $\mathcal{B}$ outputs $\pi = (\Upsilon, g, h_1, \cdots, h_n, t, T, n, N)$ as the system parameters. Clearly, due to the randomness of y_i, π has the same distribution as that in the real scheme and the simulation of the system parameters is perfect.

Public key simulation. $\mathcal{B}$ randomly chooses $k^* \in \{1, \cdots, T\}$ and denote $\overline{K^*} = \{1, \cdots, T\} \setminus \{k^*\}$. For $k \neq k^*$, $\mathcal{B}$ randomly chooses $x_k \in \mathbb{Z}_p$ and computes $X_k = e(g, g)^{x_k}$. For $k = k^*$, define $x_{k^*} = y_0 \beta^{n+1}$ which is unknown to $\mathcal{B}$. However, $\mathcal{B}$ can compute the public key for dealer k^* by computing $X_{k^*} = e(g, g)^{x_{k^*}} = e(g^\beta, g^{\beta^n})^{y_0}$. For a dealer $k \in \{1, \cdots, T\}$, her public key is X_k. The public BE key is

$$PK = \prod_{k=1}^{t} e(g,g)^{x_k} = e(g,g)^{(x_1+\cdots+x_{k^*-1}+x_{k^*+1}+\cdots+x_T)+y_0\beta^{n+1}} \overset{\texttt{Def}}{=} e(g,g)^x.$$

It is easy to see that, due to the randomness of x_k and y_0, X_k and PK are well-formed and have the same distribution as those in the real scheme. The simulation of each dealer's public key and the public broadcast encryption key[3] is also perfect.

Simulation of private keys of corrupted dealers. $\mathcal{B}$ guesses $t-1$ dealers for indices $\ell \in \tilde{\mathbb{A}}$ that $\mathcal{A}$ might corrupt. $\mathcal{B}$ follows the real scheme to generate the secret keys of the corrupted dealers for indices $\ell \in \tilde{\mathbb{A}}$, where $\tilde{\mathbb{A}}$ contains at most $t-1$ dealers. Without loss of generality, we assume that $|\tilde{\mathbb{A}}| = t-1$. Indeed, if $|\tilde{\mathbb{A}}| < t-1$, $\mathcal{B}$ can query other corrupted dealers on behalf of the attacker $\mathcal{A}$. For a corrupted dealer $\ell \in \tilde{\mathbb{A}}$, $\mathcal{B}$ randomly chooses $a_{\ell,1}, \cdots, a_{\ell,t-1} \in \mathbb{Z}_p$ and sets $f_\ell(\alpha) = x_\ell + a_{\ell,1}\alpha + \cdots + a_{\ell,t-1}\alpha^{t-1} \mod p$. If $\mathcal{A}$ queries private keys of dealers outside $\tilde{\mathbb{A}}$, $\mathbb{B}$ declares FAILURE and aborts the game. Else, the simulation is perfect. The probability of FAILURE is at most $1 - \frac{1}{C_T^{t-1}}$.

Simulation of secret key shares of corrupted dealers. Since the attacker is allowed access to the secret key shares held by a corrupted dealer $\ell \in \tilde{\mathbb{A}}$, $\mathcal{B}$ needs to simulate the key shares $S_{k,\ell}$ of the corrupted dealers $k \in \tilde{\mathbb{A}}$ and the non-corrupted dealers $k \notin \tilde{\mathbb{A}}$.

If $k \neq k^*$, $\mathcal{B}$ follows the real scheme to generate the secret key share $S_{k,\ell} = g^{f_k(\ell)}(\ell \in \tilde{\mathbb{A}}, k \neq k^*)$ which is sent to corrupted dealer ℓ by a dealer $k \neq k^*$. If $k = k^*$, $\mathcal{B}$ randomly chooses $S_{k^*,\ell} \in \mathbb{G}$.

From Shamir's secret sharing scheme (see Section 2.2 above), there exists a $(t-1)$-degree polynomial $f_{k^*}(\alpha)$ in $\mathbb{Z}_p$ such that $S_{k^*,\ell} = g^{f_{k^*}(\ell)}$. However, since the attacker can only know at most $t-1$ shares, $f_{k^*}(\ell')$ for $\ell' \notin \tilde{\mathbb{A}}$ is uniformly random from the attacker's perspective. Hence, the simulation of the secret key shares of corrupted dealers is perfect.

After simulating the corrupted dealer ℓ's secret key shares $S_{k,\ell}$ from all the other dealers, the master secret key of the corrupted dealer $\ell \in \tilde{\mathbb{A}}$ is:

$$S_\ell = S_{k^*,\ell} \prod_{k=1}^{T,k\neq k^*} S_{k,\ell} = g^{f_{k^*}(\ell)} \prod_{k=1}^{T,k\neq k^*} g^{f_k(\ell)} = g^{f_1(k)+\cdots f_T(k)},$$

which is clearly well-formed and perfectly simulated.

[3] Here, the uniform randomness of x in $\mathbb{Z}_p$ (and accordingly, the public broadcast encryption key $PK = e(g,g)^x$ in $\mathbb{G}_T$) also relies on our assumption that the corrupted dealers will follow the protocol, although they may leak their secrets and internal state information to the attacker (refer to Remark 1). If the dealers are corrupted before the key generation phase and deviate from the key generation sub-protocol, the corrupted dealers can bias $x = x_1 + x_2 + \cdots + x_T$ from uniform randomness using Gennaro et *al.*'s attack [30]. This deviation does not raise serious security flaws but leaves a gap in the simulation. This can be addressed by letting the dealers commit to their secret polynomials before the key generation phase and use the information-theoretic verifiable secret sharing scheme [31] to replace Shamir's secret sharing scheme. For more details about the attack and the fix, please refer to [30].

Decryption key share simulation. $\mathcal{B}$ can simulate the decryption key shares of users for indices $i \in [1, n] \setminus \bar{\mathbb{R}}$. Since the decryption key can be recovered from the decryption key shares, we do not need to specifically simulate the decryption key for users in $[1, n] \setminus \bar{\mathbb{R}}$.

The decryption key share can be from corrupted dealers $k \in \tilde{\mathbb{A}}$ and non-corrupted dealers $k \notin \tilde{\mathbb{A}}$. For corrupted dealers $k \in \tilde{\mathbb{A}}$, $\mathcal{B}$ does as in the real scheme. That is, $\mathcal{B}$ randomly chooses $r_{i,k} \in \mathbb{Z}_p$ and outputs the corrupted user i's decryption key share $s_{i,k} = \langle s_{i,0,k}, \cdots, s_{i,i-1,k}, s_{i,i,k}, s_{i,i+1,k}, \cdots, s_{i,n,k} \rangle = \langle g^{-r_{i,k}}, h_1^{r_{i,k}}, \cdots, h_{i-1}^{r_{i,k}}, S_j h_i^{r_{i,k}}, h_{i+1}^{r_{i,k}}, \cdots, h_n^{r_{i,k}} \rangle$.

For non-corrupted dealers $k \notin \tilde{\mathbb{A}}$, $\mathcal{B}$ also needs to simulate the corrupted user i's decryption key share:

$$s_{i,k} = \langle s_{i,0,k}, \cdots, s_{i,i-1,k}, s_{i,i,k}, s_{i,i+1,k}, \cdots, s_{i,n,k} \rangle_{k \notin \tilde{\mathbb{A}}}.$$

Here, the simulation must guarantee that, for any subset $\mathbb{A}$ with t indices in $\{1, \cdots, T\}$, a valid decryption key for corrupted user i can be reconstructed:

$$\Big(\prod_{k \in \mathbb{A}} s_{i,0,k}^{\lambda_k}, \prod_{k \in \mathbb{A}} s_{i,1,k}^{\lambda_k}, \cdots, \prod_{k \in \mathbb{A}} s_{i,n,k}^{\lambda_k}\Big).$$

$\mathcal{B}$ finishes the simulation for this case in three steps. In the first step, $\mathcal{B}$ generates a trial but valid decryption key d_i for user i. In the second step, $\mathcal{B}$ determines a trial but valid decryption key share from each non-corrupted dealer such that any t shares can be used to recover d_i. In the third step, $\mathcal{B}$ randomizes the decryption shares obtained from the second step and sends the randomized shares to the corrupted user i. After the third step, user i will recover different valid decryption keys if the user uses different sets of t shares (see Footnote 2). The third step ensures that the simulated decryption shares are distributed and work as the shares in the real world. In the following, we show how $\mathcal{B}$ can finish the three steps.

We begin with the first step. To generate a valid decryption key for corrupted user $i \in [1, n] \setminus \bar{\mathbb{R}}$, $\mathcal{B}$ randomly chooses $z_i \leftarrow \mathbb{Z}_p$ and formally sets $r_i = z_i - y_0 \beta^{n+1-i}$. It outputs

$$d_i = \langle d_{i,0}, \cdots, d_{i,n} \rangle : d_{i,0} = g^{-r_i}, d_{i,i} = g^x h_i^{r_i}, d_{i,j} = h_j^{r_i} (\forall j \neq i).$$

Clearly, d_i is well-formed and has the same distribution as that in the real scheme. Furthermore, notice that $\mathcal{B}$ can compute all these terms from the BDHE challenge instance, in particular

$$\begin{aligned} d_{i,i} = g^x h_i^{r_i} &= g^{(x_1 + \cdots + x_{k^*-1} + x_{k^*+1} + \cdots + x_T) + y_0 x^{n+1} + (y_i + \beta^i)(z_i - y_0 \beta^{n+1-i})} \\ &= g^{x_1 + \cdots + x_{k^*-1} + x_{k^*+1} + \cdots + x_T} \times g^{y_0 \beta^{n+1} + y_i z_i - y_i y_0 \beta^{n+1-i} + \beta^i z_i - y_0 \beta^{n+1}} \\ &= g^{(x_1 + \cdots + x_{k^*-1} + x_{k^*+1} + \cdots + x_T) + y_i z_i - y_i y_0 \beta^{n+1-i} + \beta^i z_i} \end{aligned}$$

which can be computed as the β^{n+1} term in the exponent cancels out. This finishes the first step.

Then we move to the second step to let $\mathcal{B}$ determine a trial but valid decryption key share from each non-corrupted dealer such that any t shares can be used to recover d_i. To this end, for $j = 0, \cdots, n$, $\mathcal{B}$ needs to find $(B_{i,j,0}, B_{i,j,1},$

$\cdots, B_{i,j,t-1}) \in \mathbb{G}^t$ to compute the function $F_{i,j}(\alpha) = B_{i,j,0}B_{i,j,1}^{\alpha} \cdots B_{i,j,t-1}^{\alpha^{t-1}}$ such that, $d_{i,j} = \prod_{k\in\mathbb{A}} s_{i,j,k}^{\lambda_k} = \prod_{k\in\mathbb{A}} F_{i,j}(k)^{\lambda_k}$. Note that $F_{i,j}(\alpha)$ is uniquely determined by any t pairs $(\alpha, F_{i,j}(\alpha))$. Specifically, $\mathcal{B}$ can use the $t-1$ pairs $(k, F_{i,j}(k))$ from corrupted dealers $k \in \tilde{\mathbb{A}}$ and the pair $(0, F_{i,j}(0)) = (0, B_{i,j,0})$ in which $F_{i,j}(0)$ is the j-th component $d_{i,j}$ of the corrupted user i's decryption key vector. Then $\mathcal{B}$ can obtain $(B_{j,0}, B_{j,1}, \cdots, B_{j,t-1})$ by solving the following t-variable linear equations in $\mathbb{G}$ for $k \in \tilde{\mathbb{A}} \cup \{0\}$: $s_{i,j,k} = F_{i,j}(k) = B_{i,j,0}B_{i,j,1}^{k} \cdots B_{i,j,t-1}^{k^{t-1}}$, where $s_{i,j,0} = B_{i,j,0} = d_{i,j}$.

After obtaining $(B_{j,0}, B_{j,1}, \cdots, B_{j,t-1})$, for a non-corrupted dealer $k \notin \tilde{\mathbb{A}}$, $\mathcal{B}$ can obtain the corrupted user i's decryption key share

$$s'_{i,k} = \langle s'_{i,0,k}, \cdots, s'_{i,i-1,k}, s'_{i,i,k}, s'_{i,i+1,k}, \cdots, s'_{i,n,k} \rangle_{k\notin\tilde{\mathbb{A}}}$$

by computing $s'_{i,j,k} = F_{i,j}(k) = B_{i,j,0}B_{i,j,1}^{k} \cdots B_{i,j,t-1}^{k^{t-1}}$ for $k \in \{1, \cdots, T\} \setminus \tilde{\mathbb{A}}$. According to Shamir's secret sharing scheme, any t shares can be used to recover d_i. This finishes the second step.

Finally, $\mathcal{B}$ moves to the third step and randomizes the computed decryption key shares. $\mathcal{B}$ randomly chooses $\delta_{i,k} \in \mathbb{Z}_p$ and randomizes $s'_{i,j,k}$ (see Footnote 1) by computing

$$s_{i,k} = \langle s'_{i,0,k}h_0^{\delta_{i,k}}, \cdots, s'_{i,i-1,k}h_{i-1}^{\delta_{i,k}}, s'_{i,i,k}h_i^{\delta_{i,k}}, s'_{i,i+1,k}h_{i+1}^{\delta_{i,k}}, \cdots, s'_{i,n,k}h_n^{\delta_{i,k}} \rangle_{k\notin\tilde{\mathbb{A}}},$$

where $h_0 \overset{\texttt{Denote}}{=} g$. This finishes the third step. Then all the corrupted users' decryption key shares from non-corrupted dealers are well-formed and have the same distribution as those in the real scheme. Note that the decryption key shares from corrupted dealers are computed as those in the real scheme. Hence, the simulation of all the decryption key shares for corrupted users is perfect.

$\mathcal{B}$ also needs to simulate $t-1$ decryption key shares for non-corrupted users. If the decryption key share is from a corrupted dealer, $\mathcal{B}$ generates it as in the real scheme. Else if the decryption key share is from a non-corrupted dealer k, since the attacker can only obtain $t-1$ decryption key shares and the Shamir's secret sharing scheme is information theoretically secure, these decryption key shares are information-theoretically independent of the decryption key of non-corrupted user i and the master secret key of non-corrupted dealer k. Hence, $\mathcal{B}$ can also follow the real scheme without knowing the master secret key. That is, $\mathcal{B}$ can perfectly simulate these decryption key shares by randomly choosing $S'_k \in \mathbb{G}, r'_{i,k} \in \mathbb{Z}_p$ and outputting user i's decryption key share

$$s_{i,k} = \langle g^{-r'_{i,k}}, h_1^{r'_{i,k}}, \cdots, h_{i-1}^{r'_{i,k}}, S'_k h_i^{r_{i,k}}, h_{i+1}^{r'_{i,k}}, \cdots, h_n^{r'_{i,k}} \rangle.$$

This finishes all the simulated services provided to the attacker.

Challenge. $\mathcal{A}$ chooses a subset $\mathbb{R}^* \subset \bar{\mathbb{R}}$. $\mathcal{B}$ sets

$$Hdr = (c_1, c_2) : c_1 = g^{\gamma}, c_2 = (\prod_{j\in\mathbb{R}^*} h_j)^{\gamma}.$$

$\mathcal{B}$ sets $sk \leftarrow Z$ and sends $\langle Hdr, sk\rangle$ to $\mathcal{A}$. Notice that $\mathcal{B}$ can compute these terms from the instance. c_1 and sk come directly from the instance. $\mathcal{B}$ can compute c_2 since it knows $\log_g(h_i)$ for all $i \in \mathbb{R}^*$; in particular,

$$c_2 = (\prod_{j\in\mathbb{R}^*} h_j)^\gamma = (\prod_{j\in\mathbb{R}^*} g^{y_j})^\gamma = (g^\gamma)^{\sum_{j\in\mathbb{R}^*} y_j}.$$

Guess. Eventually, $\mathcal{A}$ outputs a bit b'. $\mathcal{B}$ sends b' to the BDHE challenger.

Success probability. We compute the advantage of $\mathcal{B}$ to break the BDHE assumption. Assume that $\mathcal{B}$ does not declare FAILURE during the semi-adaptive game. When $b = 0$ in the semi-adaptive game, (Hdr, sk) is generated according to the same distribution as in the real world. This is also true in $\mathcal{B}$'s simulation: when $b = 0, sk = e(g,g)^{x\gamma}$, and so the challenge is a valid ciphertext under randomness γ. When $b = 1$ in the semi-adaptive game, $\langle Hdr, sk'\rangle$ is generated as in the real world, but sk' is replaced by $sk \leftarrow \mathbb{K}$, and $\langle Hdr, sk\rangle$ is sent to the attacker. This distribution is identical to that of $\mathcal{B}$'s simulation, where Hdr is valid for randomness γ, but $sk = Z$ is a uniformly random element of $\mathbb{G}_T$. From this, we see that $\mathcal{B}$'s advantage in deciding the BDHE instance is precisely $\mathcal{A}$'s advantage against FDBE if $\mathcal{B}$ does not declare FAILURE during the semi-adaptive game. By taking into account the simulation failure, $\mathcal{B}$'s advantage to break the BDHE assumption is $\varepsilon' \geq \frac{1}{C_T^{t-1}}\epsilon$.

Time complexity. The additional overhead for $\mathcal{B}$ is to simulate the answers that $\mathcal{A}$ may query. In the Setup stage, $\mathcal{B}$ needs $n+1$ exponentiations in $\mathbb{G}$. In the Public key simulation, $\mathcal{B}$ needs n exponentiations and two pairing operations to compute $e(g,g)$ and $e(g^\beta, g^{\beta^n})$. In the Simulation of private keys of corrupted dealers, $\mathcal{B}$ only needs to sample $(t-1)^2$ elements from $\mathbb{Z}_p$. This overhead is negligible, compared to the exponentiations. In the Simulation of secret key shares of corrupted dealers, $\mathcal{B}$ needs $(T-1)(t-1)$ exponentiations in $\mathbb{G}$, $O(t)$ multiplications and additions in $\mathbb{Z}_p$. The multiplications and additions can be neglected, compared to the exponentiations. In the Decryption key simulation, $\mathcal{B}$ needs at most $(n-1)(n+1)(t-1)$ exponentiations in $\mathbb{G}$ to compute the decryption key shares from the corrupted dealers, $(n-1)(n+1)$ exponentiations in $\mathbb{G}$ to compute the trial decryption key for corrupted users (there are at most $n-1$ corrupted users), $O((t-1)t^3)$ exponentiations in $\mathbb{G}$ for the $(t-1)$ t-variable linear equations, $(t-1)(n-t+1)(n+1)t$ exponentiations in $\mathbb{G}$ to compute the decryption key shares from the non-corrupted dealers, at most $(n-1)(n-t+1)(n+1)$ exponentiations in $\mathbb{G}$ to randomize these decryption key shares, and at most $(t-1)n^2$ exponentiations to simulate the decryption key shares of the non-corrupted users. In the Challenge stage, $\mathcal{B}$ needs one exponentiation in $\mathbb{G}$ to compute the challenge ciphertext. Let $\tau_{\texttt{Pair}}$ denote the overhead to compute a pairing, and $\tau_{\texttt{Exp}}$ denote the time complexity to compute one exponentiation without differentiation of exponentiations in different groups. Note that $1 \leq t \leq T$. By summing up all of them, the time complexity of $\mathcal{B}$ is $\tau' \leq \tau + \mathcal{O}(2\tau_{\texttt{Pair}} + (n^2T^2 + T^4 + n^3)\tau_{\texttt{Exp}})$. This completes the proof.