Information Sciences 502 (2019) 109–124

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Functional broadcast encryption with applications to data sharing for cloud storage

Huige Wang[a,b,*], Yuan Zhang[c,d], Kefei Chen[e,g], Guangye Sui[b], Yunlei Zhao[b], Xinyi Huang[f]

[a] *Department of Computer, Anhui Science and Technology University, Bengbu 340303, China*
[b] *Department of Computer Science and Technology, Fudan University, Shanghai, 200433, China*
[c] *Center for Cyber Security, School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China*
[d] *Department of Electrical and Computer Engineering, University of Waterloo, Waterloo N2L 3G1, Canada*
[e] *School of Science, Hangzhou Normal University, Hangzhou 310036, China*
[f] *Fujian Provincial Key Laboratory of Network Security and Cryptology, College of Mathematics and Informatics, Fujian Normal University, Fuzhou 350108, China*
[g] *Westone Cryptologic Research Center, Beijing 100070, China*

A R T I C L E   I N F O

Article history:
Received 24 January 2019
Revised 23 April 2019
Accepted 10 June 2019
Available online 10 June 2019

Keywords:
Access control
Data sharing
Functional broadcast encryption
Indistinguishable obfuscation

A B S T R A C T

Cloud storage services provide data owners an efficient and flexible way to share data. Among the shared data, some of them are very sensitive, and should be prevented for any leakage. Should users conventionally encrypt the data, however, flexibly sharing is lost. Public-key encryption with access control (PEAC) resolves this tension. Most of existing PEAC schemes only support the data owner to control either the parts of data to be accessed by other users (file-based PEAC), or the membership of users that access the entire data set (receiver-based PEAC). However, in reality a PEAC scheme with both file-based and receiver-based functionalities is required to ensure the efficiency, flexibility, and fine-grainess of the data sharing service. In this paper, we introduce a primitive of functional broadcast encryption (FBE). FBE is a manifestation of PEAC that enables a data owner to share a set of data files to a group of users, where only a specific subset of data files can be accessed and decrypted by a specific subgroup of users. We describe a construction for FBE based on indistinguishability obfuscation ($i\mathcal{O}$). Security analysis demonstrates that the proposed scheme achieves selective IND-CCA security, and a comprehensive performance analysis shows the proposed scheme is efficient.

1. Introduction

With cloud storage services, users outsource their data to cloud servers and share that data with other users without keeping online [36,38]. This enables users to remotely manage their data via the Internet and efficiently share the data with each other [23,32]. Typical applications include cloud-assisted electronic health (eHealth) systems, where a patient outsources his electronic health records (EHRs) to a cloud server and authorizes doctors in a subset to access the outsourced

* Corresponding author at: Department of Computer, Anhui Science and Technology University, Bengbu 340303, China.
E-mail addresses: whgexf@163.com (H. Wang), ZY_LoYe@126.com (Y. Zhang), kfchen@hznu.edu.cn (K. Chen), suiguangye@fudan.edu.cn (G. Sui), ylzhao@fudan.edu.cn (Y. Zhao), xyhuang@fjnu.edu.cn (X. Huang).

https://doi.org/10.1016/j.ins.2019.06.028
0020-0255/© 2019 Elsevier Inc. All rights reserved.

EHRs as needed. This can free patients and doctors from heavy local storage costs, and provides users a reliable way to store important data for a prolonged period of time.

While the deployment of cloud storage services [25,31,33–35,37] has significant benefits in sharing data for users, it also suffers from critical security challenges in data outsourcing and sharing. One of the most vital issues is data confidentiality. Among the shared data, some of them are very sensitive, and should not be leaked for preserving privacy. Therefore, users always encrypt the data before outsourcing. This can be achieved by utilizing conventional encryption, but it makes flexibly sharing inefficient. Specifically, in a cloud-assisted eHealth system, only a subset of the entire EHRs can be shared with a specific doctor during a diagnosis to prevent the irrelevant EHRs from abusing. If the entire EHR set is encrypted under the users secret key, the user has to download and decrypt the entire EHR set, agree a key with the doctor, encrypt the required EHRs under the agreed key, and outsource the ciphertexts to the cloud server. Then the doctor can access the required ciphertexts from the cloud server for the diagnosis. Considering the large size of the outsourced EHRs and the patients constrained resource capability in eHealth systems, the EHRs sharing is formidable and expensive for the patient.

To efficiently and securely share data with others, a data owner can encrypt entire data set using a system public key, and control that a target user only can decrypt a specific part of the data set or control which users can decrypt the entire data. The encryption schemes that support efficient data sharing are called public-key encryption with access control (PEAC) [29]. Existing PEAC schemes can mainly be categorized into two classes from the perspective of what it can achieve.

- File-based PEAC (FB-PEAC). For a specific user, the data owner can determine which part of the entire data set can be decrypted by the user. The key technique behind FB-PEAC is public-key functional encryption [20]. A functional encryption scheme enables a data owner to encrypt a data set and share the ciphertexts to a user such that the user can obtain a (specific) function value of the data set from the ciphertexts, but nothing more about the data set itself [3,6,15,30].
- Receiver-based PEAC (RB-PEAC). For a group of users, the data owner can control who can decrypt the entire data set. The key technique behind RB-PEAC is broadcast encryption [5,18]. In a broadcast encryption scheme, a data owner (broadcaster) encrypts a data for some subset of users who are listening on a broadcast channel. Any user in the subset can decrypt the ciphertexts while any user outside of the subset cannot.

However, in many situations, there is a need to achieve the file-based and receiver-based functionalities simultaneously. For example, in a group consultation of cloud-assisted eHealth systems, a patient is diagnosed by multiple doctors simultaneously. For the propose of preserving the patient's privacy, each doctor in the authorized subset is allowed to access a part of the entire EHR data set. A straightforward way to achieve it is to require the data owner to split EHRs into multiple parts, encrypt different parts of EHRs using different keys, and share the corresponding key to the doctor, i.e., let the patient invoke FB-PEAC or RB-PEAC repeatedly. Nevertheless, the data owner incurs a heavy burden in terms of computation and communication overhead. Nevertheless, the data owner incurs a heavy burden in terms of computation and communication overhead.

In this paper, we introduce a concept of functional broadcast encryption (FBE). FBE is a manifestation of PEAC but goes one step beyond existing PEAC schemes that achieves both file-based and receiver-based functionalities. We define the security model of FBE and argue that FBE is technically and economically viable. Furthermore, we propose a concrete FBE scheme dubbed FBRB[1], which shows the feasibility of FBE. We formally define the security model of FBE and prove the security of FBRB under the model.

Specifically, the contributions of this paper are as follows.

- We introduce the concept of functional broadcast encryption (FBE) which is a manifestation of PEAC and enables the data owner, who encrypts the entire data set and shares it with a group of users, to allow any subgroup of users to decrypt a specific part of the entire data set. We also formally define the security of FBE and demonstrate the feasibility of FBE.
- We propose a concrete FBE scheme called FBRB. FBRB is constructed on indistinguishability obfuscation ($i\mathcal{O}$), puncturable pseudorandom function (puncturable PRF), and pseudorandom generator (PRG), and achieves file-based and receiver-based functionalities simultaneously.
- We formally prove the security of FBRB under the defined model, which shows that FBRB is IND-CCA secure. We also conduct a comprehensive efficiency analysis, and demonstrate that FBRB is efficient in terms of the size of the public parameter, ciphertexts and functional keys.

The remainder of this paper is organized as follows. We discuss the related work in Section 2. In Section 3, we state the problem to be addressed. In Section 4, we present the preliminaries. Then we give the notion of functional broadcast encryption and its security model in Section 5. In Section 6, we present the construction of FBRB. We provide the security proof and performance analysis for FBRB in Sections 7 and 8, respectively. In Section 9, we present further discussion on the proposed scheme. Finally, we draw the conclusions and present some future directions in Section 10.

[1] a <u>f</u>ile-<u>b</u>ased and <u>r</u>eceiver-<u>b</u>ased scheme.

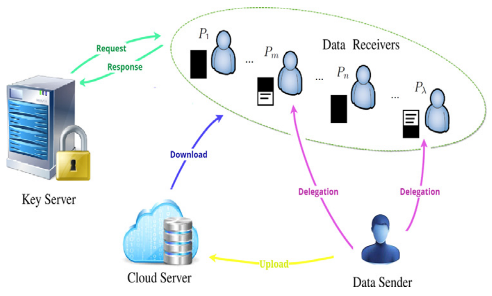

Fig. 1. Data sharing in cloud storage.

2. Related works

Data sharing service is one of the most important services in cloud storage, where a data owner is able to share his/her data with other users without keeping online. However, to protect the data owner's privacy against leakage, the data should be encrypted before outsourcing, which makes efficiently data sharing challenging. PEAC resolves this tension and provides an efficient way to share data with privacy preservation. Prior works on PEAC can mainly be classified into two categories.

- File-based PEAC (FB-PEAC) [9]. An FB-PEAC scheme can be constructed on functional encryption (FE) which enables a key holder to learn a specific function of encrypted data, but learn nothing else about the data. FE was first proposed by Katz et al. [21], then some variants were proposed [1,12,20]. More concretely, an FB-PEAC scheme enables a data owner to share a set of files with a specific user, where the data owner can control which part of the file set can be accessed by the user. This mechanism is very useful in many scenarios. For example, in a cloud-assisted eHealth system, a patient needs to share her/his outsourced EHRs with a doctor for a diagnosis, but the doctor is only allowed to access the part of the EHRs that is related to the diagnosis and cannot violate the patient's privacy by retrieving other EHRs.
- Receiver-based PEAC (RB-PEAC) [14]. A RB-PEAC scheme can be constructed on broadcast encryption (BE) which enables a data owner (broadcaster) to encrypt a data for some subset of users who are listening on a broadcast channel. Any user in the subset can decrypt the ciphertexts while any user outside of the subset cannot. The first BE scheme was proposed by Fiat and Naor [13]. Then the efficiency was enhanced by the following works [5,7,16]. therefore, based on BE, a RB-PEAC scheme enables a data owner to share a file with a group of users, where the data owner can determine the membership of the users that can access the shared data. This mechanism is widely applied many scenarios. For instance, in a cloud-assisted eHealth system, a patient needs to share her/his full outsourced EHRs with a group of doctors for a joint diagnosis. To implement this functionality, the patient first encrypts his/her EHRs along with the identity information about these doctors under this scheme and broadcast it to all users.

However, in many situations, there is a need to control that a specific part of files can only be accessed by a subgroup of users. This cannot be efficiently addressed by existing schemes.

3. Problem statement

3.1. Target problem

In many cloud-assisted data sharing systems, there is a need to allow a data owner to share a set of encrypted files with a group of users, where only authorized users can decrypt a specific part of the entire data set without requiring the data owner to keep online. For example, in cloud-assisted eHealth systems, ciphertexts of patients' EHRs are outsourced to the cloud server. When a patient wants to consult a doctor, she/he can authorize the doctor to access (decrypt) a part of her/his EHRs. When she/he needs to be diagnosed by a group of doctors (namely group consultation), each doctor only can access a specific part of the EHRs. In this paper, we target to design a PEAC scheme to address this problem.

3.2. System model

Generally, our system model includes four entities, which is shown in Fig. 1.

- Data sender. The data sender is the data owner who has a set of data to be shared. She/he first encrypts the data and outsources the ciphertexts to the cloud server. At a later point in time, she/he needs to share the data with others.
- Cloud server. The cloud server is subject to the cloud service provider. It provides the storage service and data sharing service.
- Data receiver. The data receiver obtains ciphertexts from the cloud server. To protect data owner's privacy, the data receiver only can decrypt the data that is authorized by the data owner.
- Key server. The key server is introduced to assist the data receiver in generating decryption key.

3.3. Threat model

We will consider threats from the cloud server and data receiver. Specifically, the cloud server, who may be incentivized by a malicious data receiver, attempts to violate the data sender's privacy by extracting the content from outsourced ciphertexts.

3.4. Design goals

In this paper, we target the file-based PEAC and receiver-based PEAC for cloud data sharing, in which the following objectives should be achieved.

- Functionalities: The file-based PEAC and the receiver-based PEAC should be achieved simultaneously.
- Efficiency: The communication, computation, and storage costs should be as efficient as possible.
- Security: The content of the outsourced data should not be leaked to unauthorized data receiver.

4. Notations and preliminaries

4.1. Notations

For a natural number $\mathbb{N}$, let $\lambda \in \mathbb{N}$ denote the security parameter. For a probabilistic polynomial time (PPT) algorithm A, we denote $y \leftarrow A(x_1, \dots; R)$ the process of running algorithm A on inputs $x_1, \dots$ with randomness R and outputting y as the result. For simplicity, we write $y \leftarrow A(x_1, \dots; R)$ as $y \leftarrow_\$ A(x_1, \dots)$ with implied randomness R. For an integer n, we let $[n]$ denote the set $\{1, \dots, n\}$. We say that a function $negl(\lambda)$ is negligible in λ, if $negl(\lambda) \in \lambda^{-\omega(1)}$; A function $poly(\lambda)$ is a polynomial in λ if $poly \in \lambda^{\mathcal{O}(1)}$. For a program P (or called circuit P), by $P[z]$, we emphasize that a value z is hardwired into P. For a set S, we denote $\overline{S}$ the complement of the set S.

For security proof, we follow the code-based game playing framework presented in [4,27]. A game G is defined as several procedures including an initialization procedure, some intermediate procedures and a finalization procedure. In the execution, G calls adversary A after some initialization. Adversary A is allowed to make queries permitted by game G. When A finishes all executions, G continues to execute using A's output to produce the final output of the game. For convenience, we denote $\mathsf{G}^A = y$ the event that G executes with A and outputs y. Generally, we abbreviate $\mathsf{G}^A = true$ and $\mathsf{G}^A = 1$ as G.

4.2. Puncturable pseudorandom functions

Here, we recall the notion of puncturable pseudorandom function from [8,11,22,28]. A puncturable pseudorandom function, over domain $\mathcal{D}$ and range $\mathcal{Y}$, consists of three PPT algorithms $\mathsf{PRF} = (\mathsf{PRF.K}, \mathsf{PRF.Punc}, \mathsf{PRF.Eval})$.

- $\mathsf{PRF.K}$. This is the key generation algorithm. It takes as input 1^λ, outputs a PRF key k as well as a PRF PRF_k.
- $\mathsf{PRF.Punc}$. This is the punctured key generation algorithm. It takes as input PRF_k, a set $S \subset \mathcal{D}_\lambda$, and outputs the punctured key $\mathsf{PRF}_k^{\overline{S}}$.
- $\mathsf{PRF.Eval}$. This is the evaluation algorithm. It takes as input PRF_k (or punctured key), $x \in \mathcal{D}_\lambda$, and outputs $y \in \mathcal{Y}_\lambda$. i.e., $y = \mathsf{PRF}_k(x)$ is computed on the point x using PRF_k.

Definition 1 (Security)**.** A puncturable PRF $\mathsf{PRF} = (\mathsf{PRF.K}, \mathsf{PRF.Eval}, \mathsf{PRF.Punc})$ is secure if it satisfies the following two properties.

1. Functionality preserved under puncturing. Let $\mathsf{PRF}_k \leftarrow_\$ \mathsf{PRF.K}(1^\lambda)$, $\mathsf{PRF}_k^{\overline{S}} \leftarrow \mathsf{PRF.Punc}(\mathsf{PRF}_k, S)$. Then for all $x \notin S$, we have $\mathsf{PRF}_k(x) = \mathsf{PRF}_k^{\overline{S}}(x)$.
2. preserved at punctured points. For any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ such that $\mathcal{A}_1(1^\lambda)$ defines $S \subset \mathcal{D}_\lambda$, for all $x \in S$, $\mathsf{PRF}_k \leftarrow_\$ \mathsf{PRF.K}(1^\lambda)$, $\mathsf{PRF}_k^{\overline{S}} \leftarrow \mathsf{PRF.Punc}(\mathsf{PRF}_k, S)$ and $y \leftarrow_\$ \mathcal{R}_\lambda$, there exists a negligible function $negl$ such that

$$\mathsf{Adv}^{pr}_{\mathsf{PRF},\mathcal{A}}(\lambda) = \Pr[\mathcal{A}_2(\mathsf{PRF}_k^{\overline{S}}, x, \mathsf{PRF}_k(x)) = 1] - \Pr[\mathcal{A}_2(\mathsf{PRF}_k^{\overline{S}}, x, y) = 1] \leq negl(\lambda).$$

In fact, the puncturable pseudorandom function is easily constructed by the GGM construction [19] of pseudorandom function from one-way functions.

4.3. Indistinguishability obfuscation

We follow the definition of indistinguishability obfuscation ($i\mathcal{O}$) proposed by Garg et al. [15].

Definition 2 (Indistinguishability Obfuscation)**.** A uniform PPT algorithm $i\mathcal{O}$ is called an indistinguishability obfuscator for a circuit class $\{\mathcal{C}_\lambda\}$ if the following holds:

1. Correctness. For every $\lambda \in \mathbb{N}$, every $C \in \mathcal{C}_\lambda$, every input x in the domain of C, we have:

$$\Pr\left[C'(x) = C(x) \middle| C' \leftarrow i\mathcal{O}(C)\right] = 1.$$

2. Indistinguishability. For every $\lambda \in \mathbb{N}$, for all pairs of circuits C_0, C_1, if $C_0(x) = C_1(x)$ for all inputs x, then for all PPT adversaries $\mathcal{A}$, there exists a negligible function *negl* such that:

$$\mathsf{Adv}^{i\mathcal{O}}_{i\mathcal{O},\mathcal{A}}(\lambda) = \Pr[iO(C_0(x)) = 1] - \Pr[i\mathcal{O}(C_1(x)) = 0] \le negl(\lambda).$$

4.4. Pseudorandom generator

Let $\mathsf{PRG} : \mathcal{X} \to \mathcal{Y}$ be a function that maps from the domain $\mathcal{X}$ to the range $\mathcal{Y}$. We say that PRG is a pseudorandom generator, if for all sufficient large $\lambda \in \mathbb{N}$ and all PPT adversaries $\mathcal{A}$, there exists a negligible function *negl* such that

$$\mathsf{Adv}^{pr}_{\mathsf{PRG},\mathcal{A}} = \Pr\left[\mathcal{A}(y) = 1 \middle| y \leftarrow \mathsf{PRG}(x), x \leftarrow_\$ \mathcal{X}\right] - \Pr\left[\mathcal{A}(y) = 1 \middle| y \leftarrow_\$ \mathcal{Y}\right] \le negl(\lambda).$$

5. Functional broadcast encryption (FBE)

In this section, we introduce a concept of functional broadcast encryption (FBE). Formally, an FBE scheme with respect to message space $\mathcal{M}$ and function space $\mathcal{F}$ consists of four (PPT) algorithms FBE.Setup, FBE.KeyGen, FBE.Enc and FBE.Dec:

- $\mathsf{FBE.Setup}(\lambda, N)$: This algorithm takes as input a security parameter λ and an integer N. N is an upper bound on the number of users in the system. It outputs a master public key and master secret key pair (mpk, msk).
- $\mathsf{FBE.KeyGen}(msk, i, f)$: This algorithm takes as input a master secret key msk, a user's index $i \in [N]$ and a function $f \in \mathcal{F}$, and outputs the user i's functional key sk^i_f.
- $\mathsf{FBE.Enc}(mpk, S, m)$: This algorithm takes as input a master public key mpk, a set $S \subseteq [N]$, a message $m \in \mathcal{M}$, and generates a pair (Hdr, K_S), where Hdr is a broadcast header and K_S is a shared group key. It then encrypts m under the key K_S to generate a broadcast ciphertext c_m. Finally, it outputs $ct = (S, \mathsf{Hdr}, c_m)$ as the full ciphertext.
- $\mathsf{FBE.Dec}(sk^i_f, (S, \mathsf{Hdr}, c_m))$: This algorithm takes as input a private key sk^i_f and a ciphertext $ct = (S, \mathsf{Hdr}, c_m)$, and outputs $f(m)$ or $\perp$.

Correctness. For correctness, we require that there exists a negligible function $negl(\cdot)$ such that for all sufficiently large $\lambda \in \mathbb{N}$, all message $m \in \mathcal{M}$, all $S \subset N$, all function $f \in \mathcal{F}$ and $i \in S$, it holds $\Pr[\mathsf{FBE.Dec}(\mathsf{FBE.KeyGen}(msk, i, f), \mathsf{FBE.Enc}(mpk, S, m)) = f(m)] \ge 1 - negl(\lambda)$, where $(mpk, msk) \leftarrow \mathsf{FBE.Setup}(\lambda, N)$, and the probability is taken over the random choices of all the algorithms.

Remark 1. The above definition formalizes the notion of FBE. In the setting of secret-key, an additional input, master secret key, is provided to the encryption algorithm FBE.Enc. Since its formal description and security definition are similar to that of functional broadcast encryption, we omit it here.

IND-CPA Security. We consider two security models for FBE: indistinguishable chosen-plaintext attacks (IND-CPA) and indistinguishable chosen-ciphertext attacks (IND-CCA). Here, we define the IND-CPA security. We first give its selective version, and the IND-CCA security definition is deferred to Definition 5. In IND-CPA security, the adversary is allowed to adaptively issue functional key queries with user indices and functions of its choice. After these queries, the adversary will get nothing from the challenge ciphertext (except the challenge ciphertext itself) intended for a target recipient set S^* for which the adversary $\mathcal{A}$ has no K_{S^*}. Intuitively, the notion asks that the encryptions of any two messages, should be computationally indistinguishable even allowing the adversary to query the functional keys for any adaptively chosen function $f \in \mathcal{F}$ such that $f(m_0) = f(m_1)$. Formally, for $b = 0, 1$ we denote by $\mathsf{Exp}^{adp\text{-}cpa}(b)$ the adaptive IND-CPA experiment, parameterized by the total number of parties N and an adversary $\mathcal{A}$, which is given as follows:

- Setup Phase. The challenger $\mathcal{C}$ first runs the setup algorithm $\mathsf{Setup}(\lambda, N)$ to generate the master public key and master secret key pair (mpk, msk). It then sends mpk to the adversary $\mathcal{A}$.
- Challenge Phase. The adversary $\mathcal{A}$ delivers a challenge which includes a user set $S^* \subseteq [N]$ and two messages m_0 and m_1. For $b \in \{0, 1\}$, the challenger runs $\mathsf{FBE.Enc}(mpk, S^*, m_b)$ to generate the broadcast header and group key pair $(\mathsf{Hdr}^*, K_{S^*})$, and then outputs $ct = (S^*, \mathsf{Hdr}^*, c^*)$ as the full ciphertext to users in S^*, where c^* is the encryption of m_b under K_{S^*}. Finally, the challenger sends $ct = (S^*, \mathsf{Hdr}^*, c^*)$ to the adversary.
- Functional Key Query Phase. The adversary adaptively queries the challenger with any user index $i \in [N]$ and function $f \in \mathcal{F}$ of its choice such that $f(m_0) = f(m_1)$. For each such query, the challenger replies with $sk^i_f \leftarrow \mathsf{FBE.KeyGen}(msk, i, f)$.
- Output Phase. The adversary outputs a bit b' which is defined as the output of the experiment.

We define the advantage that an adversary $\mathcal{A}$ wins in the experiment $\mathsf{Exp}^{adp\text{-}cpa}(b)$ as $\mathsf{Adv}^{adp\text{-}cpa}_{\mathsf{FBE},\mathcal{A}}(\lambda) = \Pr[\mathsf{Exp}^{adp\text{-}cpa}(0) = 1] - \Pr[\mathsf{Exp}^{adp\text{-}cpa}(1) = 1]$. In the following, we give the formal security definition.

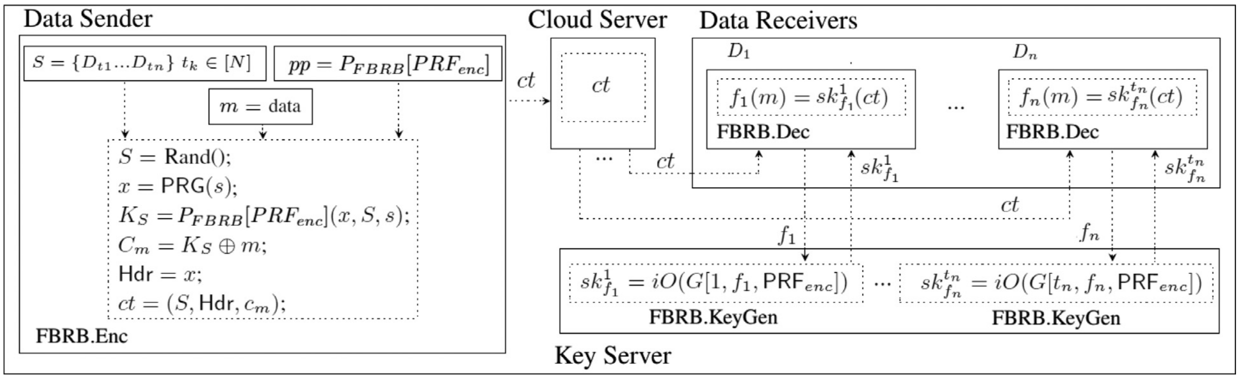
Data Sender
$S = \{D_{t1} \ldots D_{tn}\}\ t_k \in [N]$
$pp = P_{FBRB}[PRF_{enc}]$
$m = \text{data}$
$S = \text{Rand}();$
$x = \text{PRG}(s);$
$K_S = P_{FBRB}[PRF_{enc}](x, S, s);$
$C_m = K_S \oplus m;$
$\text{Hdr} = x;$
$ct = (S, \text{Hdr}, c_m);$
FBRB.Enc
Cloud Server
ct
Data Receivers
D_1
D_n
$f_1(m) = sk^1_{f_1}(ct)$
$f_n(m) = sk^{t_n}_{f_n}(ct)$
FBRB.Dec
$sk^1_{f_1}$
$sk^{t_n}_{f_n}$
f_1
f_n
$sk^1_{f_1} = iO(G[1, f_1, \text{PRF}_{enc}])$
$sk^{t_n}_{f_n} = iO(G[t_n, f_n, \text{PRF}_{enc}])$
FBRB.KeyGen
Key Server

Fig. 2. The integrated algorithms for the FBRB.

Definition 3 (Adaptive IND-CPA Security)**.** An FBE scheme FBE is **adaptive** IND-CPA secure if for any polynomial N and any PPT adversary $\mathcal{A}$, the advantage function $\mathsf{Adv}^{adp\text{-}cpa}_{\mathsf{FBE},\mathcal{A}}(\lambda)$ defined above is negligible in the security parameter λ.

Compared with adaptive IND-CPA, the selective version requires that the adversary $\mathcal{A}$ must specify the challenge set S^* and challenge messages m_0, m_1 at the beginning of the experiment. We give the formal definition by the following experiment $\mathsf{Exp}^{sel\text{-}cpa}(b)$, which is parameterized by the total number of recipient N and an adversary $\mathcal{A}$.

- Setup. The adversary $\mathcal{A}$ commits to a challenge which includes a recipient set $S^* \subseteq [N]$ and two messages m_0 and m_1. The challenger $\mathcal{C}$ runs the setup algorithm $\mathsf{Setup}(\lambda, N)$ to generate the master public key and master secret key pair (mpk, msk).
- Challenge. For $b \in \{0, 1\}$, the challenger runs $\mathsf{FBE.Enc}(mpk, S^*, m_b)$ to generate the broadcast header and group key pair $(\mathsf{Hdr}^*, K_{S^*})$ and then outputs $ct = (S^*, \mathsf{Hdr}^*, c^*)$ as the full ciphertext to users in S^*, where c^* is the encryption of m_b under the key K_{S^*}. Finally, the challenger sends the ciphertext $(S^*, \mathsf{Hdr}^*, c^*)$ and public key mpk to the adversary.
- Functional Key Query. The adversary adaptively queries the challenger with any recipient index $i \in [N]$ and function $f \in \mathcal{F}$ of its choice such that $f(m_0) = f(m_1)$. For each such query, the challenger replies with $sk^i_f \leftarrow \mathsf{FBE.KeyGen}(msk, i, f)$.
- Output Phase. The adversary outputs a bit b' which is defined as the output of the experiment.

We define the advantage that the adversary wins in the experiment $\mathsf{Exp}^{sel\text{-}cpa}(b)$ to be $\mathsf{Adv}^{sel\text{-}cpa}_{\mathsf{FBE},\mathcal{A}}(\lambda) = \Pr[\mathsf{Exp}^{sel\text{-}cpa}(0) = 1] - \Pr[\mathsf{Exp}^{sel\text{-}cpa}(1) = 1]$.

Definition 4 (Selective IND-CPA Security)**.** An FBE scheme FBE is **selective** IND-CPA secure if for any polynomial N and any PPT adversary $\mathcal{A}$, the advantage function $\mathsf{Adv}^{sel\text{-}cpa}_{\mathsf{FBE},\mathcal{A}}(\lambda)$ defined above is negligible in the security parameter λ.

Remark 2. Since in the above two security definitions, we allow any user including those in the challenge set S^* to make functional key queries as long as the functions f delivered by the adversary satisfy $f(m_0) = f(m_1)$, we may consider the set S^* and message pair (m_0, m_1) together as the challenge, rather than care the position that S^* appears. Thus, compared with the security definition for broadcast encryption in [10], our definition here is more simple at the level of broadcast functionalities.

6. Proposed FBRB

In this section, we propose a manifestation of FBE called FBRB.

6.1. Scheme details

A data sender $\mathcal{P}$, a cloud server $\mathcal{CS}$, a key server $\mathcal{KS}$, a set of data receiver $\{\mathcal{D}_1, \ldots, \mathcal{D}_N\}$ are involved in FBRB. we denote a subset of data receivers by $\{\mathcal{D}_{t1}, \ldots, \mathcal{D}_{tn}\}$, where $ti \in [N]$ for all $i \in [n]$. In addition, we use $\mathcal{F}$ to denote a family of functions from which the system allows a doctor to choose one or more. The integrated algorithms of the FBRB is shown in Fig. 2 where we do not include the setup phase since the parameter generated in it is global.

Let $\mathsf{PRG} : \{0,1\}^\lambda \to \{0,1\}^{2\lambda}$ be a pseudorandom generator, $\{\mathsf{PRF}_k : \{0,1\}^{2\lambda} \times 2^{[N]} \to \mathcal{K}\}_{k \in \mathbb{N}}$ be a puncturable PRF family, $\mathcal{K}$ be a group key space. Our scheme FBRB is constructed over the message space $\mathcal{M} = \{0,1\}^{\mathsf{m\text{-}len}(\lambda)}$, the ciphertext space $\mathcal{CP} = 2^{[N]} \times \{0,1\}^{2\lambda} \times \{0,1\}^{\mathsf{m\text{-}len}(\lambda)}$ and the function space $\mathcal{F}$, where $\mathsf{m\text{-}len}$ is a polynomial in the security parameter λ and $\mathcal{F}$ is a function space which includes any one-way functions except constant functions and identity functions.

- Setup Phase. With the security parameter λ and the total number N of the data receivers allowed in this system, the system first run $\mathsf{FBRB.Setup}$ to generate the master secret key and master public key. Specifically, the algorithm

- **Constants** : PRF_{enc}
- **Input** : x, S, s

1. If $\mathsf{PRG}(s) \neq x$, output $\perp$
2. Otherwise, output $\mathsf{PRF}_{enc}(x, S)$.

Fig. 3. Circuit $P_{\mathsf{FBRB}}[\mathsf{PRF}_{enc}]$.

- **Constants** : i, f, PRF_{enc}
- **Input** : S, x, c_m

1. If $i \notin S$, output $\perp$
2. Otherwise, compute $K_S = \mathsf{PRF}_{enc}(x, S)$
3. Compute $m = K_S \oplus c_m$
4. Output $y = f(m)$

Fig. 4. Circuit $G[i, f, \mathsf{PRF}_{enc}]$.

first samples $\mathsf{PRF}_{enc} \leftarrow \mathsf{PRF.K}(1^\lambda)$ and then constructs a circuit $P_{\mathsf{FBRB}}[\mathsf{PRF}_{enc}]$ as in Fig. 3. Next, it computes $P = iO(P_{\mathsf{FBRB}}[\mathsf{PRF}_{enc}])$. The master secret key is $msk = \mathsf{PRF}_{enc}$ and the master public key is $mpk = P$.

- Outsourcing Phase. With the master public key mpk, a data m, the data sender $\mathcal{P}$ first chooses a subset $S \in [N]$ of data receivers. Then, he/she runs $\mathsf{FBRB.Enc}$ to generate a ciphertext. Specifically, the data sender first samples a random PRG seed $s \leftarrow_\$ \{0, 1\}^\lambda$, and then computes $x = \mathsf{PRG}(s)$ and $K_S = P(x, S, s)$. Next, the data sender computes $c_m = K_S \oplus m$ and $\mathsf{Hdr} = x$ and the encryption of the data m is $ct = (S, \mathsf{Hdr}, c_m)$. Finally, the data sender outsources and uploads the ciphertext ct to the cloud server.
- Key Generation Phase. With the master secret key $msk = \mathsf{PRF}_{enc}$, when the key server receives a data receiver's identity $i \in \mathbb{N}$, and a function $f \in \mathcal{F}$ from the data receiver, the key server runs $\mathsf{FBRB.KeyGen}$ to generate a private key. Concretely, it first constructs a circuit $G[i, f, \mathsf{PRF}_{enc}]$ as in Fig. 4, then computes the private key $sk^i_f = iO(G[i, f, \mathsf{PRF}_{enc}])$ and sends sk^i_f to the data receiver i.
- Data Sharing Phase. After a data receiver requests a private key sk^i_f from the key server, where i denotes his/her identity and f is a function, the data receiver first downloads the encrypted data $ct = (S, \mathsf{Hdr} = x, c_m)$ from the cloud server, and then he/she runs $\mathsf{FBRB.Dec}$ to get the partial data that he/she needs. Concretely, the data receiver computes $f(m) = sk^i_f(S, x, c_m)$ and obtains the partial data $f(m)$ if the data receiver identity belongs to the authorized subset S encrypted in ct, namely $i \in S$.

The correctness of FBRB is proven as follows. For all the master secret key $msk = \mathsf{PRF}_{enc}$ and the master pubic key $mpk = iO(P_{\mathsf{FBRB}}[\mathsf{PRF}_{enc}])$ generated in the beginning of the scheme, all private key $sk^i_f = iO(G[i, f, \mathsf{PRF}_{enc}])$ generated by the key server, for all the ciphertext $ct = (S, \mathsf{Hdr} = x, c_m)$ generated by the data sender, where $x = \mathsf{PRG}(s)$, $\mathsf{Hdr} = x$, $S \subseteq N$, $K_S = iO(P_{\mathsf{FBRB}}[\mathsf{PRF}_{enc}])(x, S, s)$, $c_m = K_S \oplus m$ and $i \in S$, we check that the data receiver can recover the correct result $f(m)$, i.e., $\mathsf{FBRB.Dec}(sk^i_f, S, \mathsf{Hdr}, c_m) = f(m)$.

Since FBRB is an instance of the primitive of FBE, the security of the FBRB follows the security notion IND-CPA and IND-CCA of the FBE described in Section 5. The formal security proof is provided in Section 7.

7. Security of the FBRB

7.1. Overview of the security proof

We show that FBRB is secure in the IND-CPA model: an adversary, even if allowed to adaptively obtain a polynomial functional key, is still incapable of distinguishing encryptions of two challenge messages of its choice. To prove the IND-CPA security, there are two core points to care. The first is how to prevent the adversary from employing the master public key to compute the challenge shared key which is used for encrypting the challenge message. The second is, when this key becomes a random value, how to guarantee that the adversary is still able to decrypt challenge ciphertext.

Let (S^*, x^*, c^*) be a challenge ciphertext and K_{S^*} be a challenge group key generated in the challenge phase. Interestingly, the challenger can directly use the master secret key PRF_{enc} to generate the challenge ciphertext before the master public key is generated. Recall the functionality of the master public key $mpk = i\mathcal{O}(P_{\mathsf{FBRB}}[\mathsf{PRF}_{enc}])$ which is in fact an obfuscated circuit: given an input (x, S, s), this program begins with checking whether x is an image of the PRG on the seed s and ends with computing the group key $K_S \leftarrow \mathsf{PRF}_{enc}(x, S)$ and outputting K_S. To prevent the adversary from computing K_{S^*} via $mpk = i\mathcal{O}(P_{\mathsf{FBRB}}[\mathsf{PRF}_{enc}])$, we change (x^*, S^*, s^*) into an invalid input so as to make it be computationally indistinguishable from the original one. Fortunately, by the property of PRG, we can do this. Roughly, we first replace the image x^* of the PRG with a random one, and then puncture $P_{\mathsf{FBRB}}[\mathsf{PRF}_{enc}]$ on point (x^*, S^*, s^*) along with the punctured PRF key (note that the punctured key is computed on point (x^*, S^*, s^*)) hardwired in. When taking the point (x^*, S^*, s^*) as an input, the circuit will output a failure symbol, due to x^* is no longer an image of the PRG. Particularly, on other points, the circuit is still running normally.

To solve the second problem, we again use the property of the punctured PRF and the security of the $i\mathcal{O}$. Review the functionality of a functional key $i\mathcal{O}(G[i, f, \mathsf{PRF}_{enc}])$ which is in fact an obfuscated circuit, where i is a user's index (or viewed as identity) and f is a function: given an input (S, x, c_m), this circuit starts with checking whether $i \in S$ and computing a group key $K_S \leftarrow \mathsf{PRF}_{enc}(x, S)$, and ends with computing and outputting $f(m)$ by decrypting c_m with the key K_S. Subsequently, we continue the following steps. Next, we change K_{S^*} into a random one. Before this, we puncture the above circuit on the point (S^*, x^*, c^*) with the tuple (S^*, x^*, c^*) and precalculated key K_{S^*} hardwired in. When the adversary tries to decrypt (S^*, x^*, c^*) using its queried key sk_f^i, we use the hardwired key K_{S^*} directly to decrypt c^*. Thus, we reduce the gap to the security of the puncturable PRF. In addition, using the general transformation proposed by Ananth et al. [2] to our selective IND-CPA secure scheme, we may obtain the adaptive IND-CPA version.

We also prove that our scheme achieves the selective IND-CCA security. However, to prove the IND-CCA security is not trivial, since the reduction (also called the PRF adversary), which acts as the challenger of the IND-CCA adversary, has no the PRF key PRF_{enc} that is used for simulating the decryption queries. Luckily, the reduction can request the punctured key (note that this punctured key will be constructed on point (x^*, S^*)) which may be also used to simulate the decryption queries.

7.2. Selective IND-CPA security

Theorem 1. *If* PRG *is a secure pseudorandom generator,* PRF_{enc} *is a secure puncturable pseudorandom function,* $i\mathcal{O}$ *is a secure indistinguishable obfuscator, then FBRB is a* ***selective IND-CPA*** *secure functional broadcast encryption associated with the function space* $\mathcal{F}$ *and message space* $\mathcal{M}$.

Proof. We prove this theorem by a sequence of games starting with game G_0 where the challenge bit $b = 0$, ending with game G_{10} where $b = 1$. Then we prove that any two adjacent games are computationally indistinguishable. In the proof, the core idea is how to implement the conversion from challenge ciphertext which encrypts m_0 in game G_0 to challenge ciphertext which encrypts m_1 in game G_{10}. These can be achieved by puncturing PRF_{enc} at the point of challenge ciphertext, hardwiring the challenge ciphertext and punctured key into the functional key circuit. Then, we prove the punctured circuit is computationally indistinguishable from the original one. The formal proof is given below.

G_0: This is the standard selective IND-CPA game in the case $b = 0$ where the adversary $\mathcal{A}$ commits to a challenge set S^* and two challenge messages m_0, m_1 before the master public key is generated. Then, the challenger calls $\mathsf{FBRB.Setup}(\lambda, N)$ to generate a key pair (mpk, msk), where $mpk = P = iO(P_{\mathsf{FBRB}}[\mathsf{PRF}_{enc}])$ and $msk = \mathsf{PRF}_{enc}$. Next, it builds the challenge ciphertext as in Section 6: It first samples a random PRG seed $s^* \leftarrow_\$ \{0, 1\}^\lambda$. Then, it computes $x^* = \mathsf{PRG}(s^*)$, $K_{S^*} = P(x^*, S^*, s^*)$ and $c^* = K_{S^*} \oplus m_0$, and the final ciphertext is set as $ct^* = (S^*, \mathsf{Hdr}^*, c^*)$, where $\mathsf{Hdr}^* = x^*$.

G_1: This game is the same as G_0 except that the way the challenge ciphertext is generated. We construct the challenge ciphertext $ct^* = (S^*, \mathsf{Hdr}, c^*)$ as follows. The challenger $\mathcal{C}$ first samples a random $s^* \leftarrow_\$ \{0, 1\}^\lambda$ and computes $x^* = \mathsf{PRG}(s^*)$, and then computes $K_{S^*} = \mathsf{PRF}_{enc}(x^*, S^*)$ using $msk = \mathsf{PRF}_{enc}$. Next, it computes $c^* = K_{S^*} \oplus m_0$ and $\mathsf{Hdr}^* = x^*$. In fact, albeit the challenge ciphertext is generated in a different manner, it is still identical to that in game G_0.

G_2: This game is the same as G_1 except the way we generate the challenge ciphertext. When generating the challenge ciphertext $ct^* = (S^*, \mathsf{Hdr}^* = x^*, c^*)$, the challenger samples a random $x^* \leftarrow_\$ \{0, 1\}^{2\lambda}$ instead of computing $x^* \leftarrow \mathsf{PRG}(s^*)$. Obviously, this game is computationally indistinguishable from game G_1 by the security of PRG.

G_3: This game is the same as G_2 except the way we generate the master public key mpk. Let $ct^* = (S^*, x^*, c^*)$ be the challenge ciphertext. The master public key mpk is constructed as follows. First, it computes a punctured key $\mathsf{PRF}_{enc}^{\overline{\{x^*, S^*\}}} \leftarrow \mathsf{PRF.Punc}(\mathsf{PRF}_{enc}, (x^*, S^*))$ and then uses it to compute the master public key $mpk \leftarrow i\mathcal{O}(P'_{\mathsf{FBRB}})$, where the circuit P'_{FBRB} is described in Fig. 5. In addition, note that the circuit P'_{FBRB} has the punctured key $\mathsf{PRF}_{enc}^{\overline{\{x^*, S^*\}}}$ hardwired in.

G_4: This game is the same as G_3 except the way the functional key queries made by the adversary are answered. Let $ct^* = (S^*, x^*, c^*)$ denote the challenge ciphertext. When the adversary $\mathcal{A}$ makes a functional key query (i, f) such that

```
• Constants : $\mathsf{PRF}_{enc}^{\overline{\{x^*,S^*\}}}$
• Input : $x$, $S$, $s$
1. If $\mathsf{PRG}(s) \neq x$, output $\perp$
2. Otherwise, output $\mathsf{PRF}_{enc}^{\overline{\{x^*,S^*\}}}(x, S)$.
```

Fig. 5. Circuit P'_{FBRB}.

```
• Constants : $i$, $f$, $\mathsf{PRF}_{enc}^{\overline{\{x^*,S^*\}}}$, $x^*$, $S^*$, $K_{S^*}$, $c^*$
• Input : $S$, $x$, $c_m$
1. If $i \notin S$, output $\perp$
2. Otherwise if $(x, S) = (x^*, S^*)$ compute $m = K_{S^*} \oplus c^*$ //Note that this case implies $c_m = c^*$
3. Otherwise compute $K_S = \mathsf{PRF}_{enc}^{\overline{\{x^*,S^*\}}}(x, S)$
4. Compute $m = K_S \oplus c_m$
5. Output $f(m)$
```

Fig. 6. Circuit G'.

$f(m_0) = f(m_1)$, the challenger first computes a punctured PRF key $\mathsf{PRF}_{enc}^{\overline{\{x^*,S^*\}}} \leftarrow \mathsf{PRF.Punc}(\mathsf{PRF}_{enc}, (x^*, S^*))$, and then uses it to compute the private key $sk_f^i \leftarrow i\mathcal{O}(G')$, where the circuit G' is described in Fig. 6. In addition, note that the circuit G' has the values i, f, the punctured key $\mathsf{PRF}_{enc}^{\overline{\{x^*,S^*\}}}$, the challenge ciphertext S^*, x^*, c^* and the key K_{S^*} hardwired in.

G_5: This game is the same as G_4 except that we use a truly random $K_{S^*} \leftarrow_\$ \mathcal{K}$ instead of $K_{S^*} \leftarrow \mathsf{PRF}_{enc}(x^*, S^*)$ to compute the challenge ciphertext $ct^* = (S^*, x^*, c^*)$.

G_6: This game is the same as G_5 except that the challenge ciphertext encrypts the message m_1 instead of m_0.

G_7: This game is the same as G_6 except that in the generation of the challenge ciphertext, we use $K_{S^*} \leftarrow \mathsf{PRF}_{enc}(x^*, S^*)$ instead of a random $K_{S^*} \leftarrow_\$ \mathcal{K}$ to encrypt m_1, where x^* is a random value from $\{0, 1\}^{2\lambda}$.

G_8: This game is the same as G_7 except that we change the way that the functional key queries are answered. For every functional key query (i, f), we compute $sk_f^i \leftarrow i\mathcal{O}(G)$ instead of $sk_f^i \leftarrow i\mathcal{O}(G')$.

G_9: This game is the same as G_8 except that we change the manner that the challenge ciphertext $ct^* = (S^*, \mathsf{Hdr}^* = x^*, c^*)$ is generated, we compute $x^* \leftarrow \mathsf{PRG}(s^*)$ instead of choosing a random $x^* \leftarrow_\$ \{0, 1\}^{2\lambda}$.

G_{10}: This game is the same as G_9 except that we change the manner that the master public key mpk is generated. We compute mpk as $mpk \leftarrow i\mathcal{O}(P_{\mathsf{FBRB}})$ instead of $mpk \leftarrow i\mathcal{O}(P'_{FBE})$. □

Lemma 1. *Let* $\Pr[\mathsf{G}_0]$ *and* $\Pr[\mathsf{G}_1]$ *respectively denote the probability that an adversary* $\mathcal{A}$ *wins in game* G_0 *and game* G_1. *Then we have*

$$\Pr[\mathsf{G}_0] = \Pr[\mathsf{G}_1]. \tag{1}$$

Proof. The difference between G_0 and G_1 is that in G_0, the challenge ciphertext is generated according to the standard construction where the master public key is computed as $mpk = i\mathcal{O}(P_{\mathsf{FBRB}}[\mathsf{PRF}_{enc}])$ and the group key is computed as $K_{S^*} \leftarrow i\mathcal{O}(P_{\mathsf{FBRB}}[\mathsf{PRF}_{enc}])(x^*, S^*, s^*)$; while in G_1, the group key is computed in a way such that $K_{S^*} \leftarrow \mathsf{PRF}_{enc}(x^*, S^*)$. It is easy to see that, by the correctness of the scheme, the resulting key K_{S^*} in the two games are identical. So the Claim holds. □

Lemma 2. *Let* $\Pr[\mathsf{G}_1]$ *and* $\Pr[\mathsf{G}_2]$ *respectively denote the probability that an adversary* $\mathcal{A}$ *wins in game* G_1 *and game* G_2. *Then there exists an adversary* $\mathcal{A}_{prg}$ *which breaks the security of the* PRG.

$$\Pr[\mathsf{G}_1] - \Pr[\mathsf{G}_2] \leq \mathsf{Adv}_{\mathsf{PRG},\mathcal{A}_{prg}}^{pr}. \tag{2}$$

Proof. We show that if there exists a PPT adversary $\mathcal{A}$ that can distinguish G_1 from G_2, then we can use $\mathcal{A}$ to construct a PPT adversary $\mathcal{A}_{prg}$ to break the security of the PRG. Let $\mathcal{B}$ denote the challenger of the PRG, then the adversary $\mathcal{A}_{prg}$ is constructed as follows.

1. The adversary $\mathcal{A}_{prg}$ first samples a random $\mathsf{PRF}_{enc} \leftarrow \mathsf{FBRB.Setup}(\lambda, N)$, then constructs a circuit $P_{\mathsf{FBRB}}[\mathsf{PRF}_{enc}]$ and computes the master public key $mpk = i\mathcal{O}(P_{\mathsf{FBRB}}[\mathsf{PRF}_{enc}])$ and master secret key $msk = \mathsf{PRF}_{enc}$.
2. For the challenge query (S^*, m_0, m_1) from $\mathcal{A}$, the adversary $\mathcal{A}_{prf}$ first queries its challenger $\mathcal{B}$, and from which it receives x^*, where $x^* \leftarrow_\$ \{0,1\}^{2\lambda}$ or $x^* \leftarrow \mathsf{PRG}(s^*)$, $s^* \leftarrow_\$ \{0,1\}^{\lambda}$. Then, it computes $K_{S^*} = \mathsf{PRF}_{enc}(x^*, S^*)$ and $c^* = K_{S^*} \oplus m_0$. Finally, it sends the challenge ciphertext $ct^* = (S^*, \mathsf{Hdr}^*, c^*)$ and master public key mpk to $\mathcal{A}$, where $\mathsf{Hdr}^* = x^*$.
3. For the functional key query (i, f) from $\mathcal{A}$, the adversary $\mathcal{A}_{prf}$ behaves in the same manner as in game G_1 and G_2. More specifically, $\mathcal{A}_{prg}$ first computes a circuit $G[i, f, \mathsf{PRF}_{enc}]$, and then sets the functional key as $sk_f^i \leftarrow i\mathcal{O}(G[i, f, \mathsf{PRF}_{enc}])$, and sends sk_f^i to $\mathcal{A}$.
4. Finally the adversary $\mathcal{A}$ outputs a bit b' which is defined as the output of the experiment.

Clearly, when x^* is a random value, $\mathcal{A}_{prg}$ simulates game G_2 for $\mathcal{A}$, otherwise, it simulates game G_1 for $\mathcal{A}$. Thus, we have $\Pr[\mathsf{G}_1] - \Pr[\mathsf{G}_2] \le \mathsf{Adv}^{pr}_{\mathsf{PRG},\mathcal{A}_{prg}}$. □

Lemma 3. *Let* $\Pr[\mathsf{G}_2]$ *and* $\Pr[\mathsf{G}_3]$ *respectively denote the probability that an adversary* $\mathcal{A}$ *wins in game* G_2 *and game* G_3. *Then there exists an adversary* $\mathcal{A}_{io}$ *which breaks the security of the* $i\mathcal{O}$.

$$\Pr[\mathsf{G}_2] - \Pr[\mathsf{G}_3] \le \mathsf{Adv}^{i\mathcal{O}}_{i\mathcal{O},\mathcal{A}_{i\mathcal{O}}}. \tag{3}$$

Proof. The difference between games G_2 and G_3 lies in that in the former, we output $i\mathcal{O}(P_{\mathsf{FBRB}})$ as the master public key, in the latter, we output $i\mathcal{O}(P'_{FBE})$ as the master public key. The computational indistinguishability of G_2 and G_3 will be proved based on the security of $i\mathcal{O}$. Let $\mathcal{A}_{i\mathcal{O}}$ be an adversary that breaks the security of the $i\mathcal{O}$. Now $\mathcal{A}_{i\mathcal{O}}$ obtains an obfuscated circuit from its challenger $\mathcal{B}$. $\mathcal{A}_{i\mathcal{O}}$ aims to tell which circuit is obfuscated. Subsequently, we construct $\mathcal{A}_{i\mathcal{O}}$ as follows.

1. In this phase, the adversary $\mathcal{A}_{i\mathcal{O}}$ constructs the challenge ciphertext before obtaining mpk. Concretely, $\mathcal{A}_{i\mathcal{O}}$ first samples a $\mathsf{PRF}_{enc} \leftarrow \mathsf{FBRB.Setup}(\lambda, N)$ and a random $x^* \leftarrow_\$ \{0,1\}^{2\lambda}$, then computes the group key $K_{S^*} \leftarrow \mathsf{PRF}_{enc}(x^*, S^*)$ and the ciphertext $c^* = K_{S^*} \oplus m_0$. Next it computes the punctured PRF key $\mathsf{PRF}_{enc}^{\overline{\{x^*,S^*\}}} \leftarrow \mathsf{PRF.Punc}(\mathsf{PRF}_{enc}, S^*)$ and uses it to construct circuits $P_{\mathsf{FBRB}}[\mathsf{PRF}_{enc}]$ and $P'_{\mathsf{FBRB}}[\mathsf{PRF}_{enc}^{\overline{\{x^*,S^*\}}}]$. After that, $\mathcal{A}_{i\mathcal{O}}$ delivers the two circuits to its challenger $\mathcal{B}$, and from which it gets an obfuscated circuit P, where P is either the obfuscation of circuit $P_{\mathsf{FBRB}}[\mathsf{PRF}_{enc}]$ or an obfuscation of circuit $P'_{\mathsf{FBRB}}[\mathsf{PRF}_{enc}^{\overline{\{x^*,S^*\}}}]$. Finally, it sends the challenge ciphertext $ct^* = (S^*, \mathsf{Hdr}^*, c^*)$ and the master public key mpk to $\mathcal{A}$, where $\mathsf{Hdr}^* = x^*$.
2. For the functional key query (i, f) from $\mathcal{A}$, the adversary $\mathcal{A}_{i\mathcal{O}}$ computes the functional key as in games G_2 and G_3, namely $sk_f^i \leftarrow i\mathcal{O}(G[i, f, \mathsf{PRF}_{enc}])$.
3. Finally the adversary $\mathcal{A}$ outputs a bit b' which is defined as the output of the experiment.

Since x^* is a truly random value, with overwhelming probability, x^* is not in the image of the PRG. So, the circuits $P_{\mathsf{FBRB}}[\mathsf{PRF}_{enc}]$ and $P'_{\mathsf{FBRB}}[\mathsf{PRF}_{enc}^{\overline{\{x^*,S^*\}}}]$ will have identical input-output behavior. In addition, since $P = i\mathcal{O}(P_{\mathsf{FBRB}}[\mathsf{PRF}_{enc}])$ is the case in game G_2, while $P = i\mathcal{O}(P'_{\mathsf{FBRB}}[\mathsf{PRF}_{enc}^{\overline{\{x^*,S^*\}}}])$ is the case in game G_3. Thus, the difference between games G_2 and G_3 can be bounded by the security of the $i\mathcal{O}$, i.e., $\Pr[\mathsf{G}_2] - \Pr[\mathsf{G}_3] \le \mathsf{Adv}^{i\mathcal{O}}_{i\mathcal{O},\mathcal{A}_{i\mathcal{O}}}$. □

Lemma 4. *Let* $\Pr[\mathsf{G}_3]$ *and* $\Pr[\mathsf{G}_4]$ *respectively denote the probability that an adversary* $\mathcal{A}$ *wins in game* G_3 *and game* G_4. *Then for any polynomial* q_{sk}, *there exists an adversary* $\mathcal{A}'_{i\mathcal{O}}$ *that can break the security of the* $i\mathcal{O}$.

$$\Pr[\mathsf{G}_3] - \Pr[\mathsf{G}_4] \le q_{sk}.\mathsf{Adv}^{i\mathcal{O}}_{i\mathcal{O},\mathcal{A}'_{i\mathcal{O}}}. \tag{4}$$

q_{sk} *denotes the maximal number of functional key queries made by the adversary* $\mathcal{A}$.

Proof. Here we assume that in the q_{sk} functional key queries (i, f) made by the adversary $\mathcal{A}$, all functions f are different with each other. Now we consider q_{sk} intermediate hybrids $\mathsf{G}_{3,i}$ for $0 \le i \le q_{sk}$. In $\mathsf{G}_{3,i}$, we respond to the first $q_{sk} - i$ functional key queries as in game G_3 and the remaining i functional key queries as in game G_4. We show that if there exists a PPT adversary $\mathcal{A}$ that can distinguish $\mathsf{G}_{3,i}$ and $\mathsf{G}_{3,i+1}$, then we can build an adversary $\mathcal{A}'_{i\mathcal{O}}$ which can break the security of the $i\mathcal{O}$. Let $\mathcal{B}$ be the challenger of the $i\mathcal{O}$, then the adversary $\mathcal{A}'_{i\mathcal{O}}$ is constructed as follows.

1. Similar to Claim 3, the adversary $\mathcal{A}'_{i\mathcal{O}}$ also construct the challenge ciphertext for the tuple (S^*, m_0, m_1) delivered by the adversary $\mathcal{A}$ before the master public key is generated. Concretely, $\mathcal{A}'_{i\mathcal{O}}$ first samples $\mathsf{PRF}_{enc} \leftarrow \mathsf{FBRB.Setup}(\lambda, N)$ and $x^* \leftarrow_\$ \{0,1\}^{2\lambda}$, then computes the group key $K_{S^*} \leftarrow \mathsf{PRF}_{enc}(x^*, S^*)$ and the ciphertext $c^* = K_{S^*} \oplus m_0$. Next it computes the punctured key $\mathsf{PRF}_{enc}^{\overline{\{x^*,S^*\}}} \leftarrow \mathsf{PRF.Punc}(\mathsf{PRF}_{enc}, (x^*, S^*))$, constructs the circuit $P'_{\mathsf{FBRB}}[\mathsf{PRF}_{enc}^{\overline{\{x^*,S^*\}}}]$ and sets the master public key $mpk = i\mathcal{O}(P'_{\mathsf{FBRB}}[\mathsf{PRF}_{enc}^{\overline{\{x^*,S^*\}}}])$ and the master secret key $msk = \mathsf{PRF}_{enc}$. Finally, it sends the challenge ciphertext $ct^* = (S^*, \mathsf{Hdr}^*, c^*)$ and the master public key mpk to $\mathcal{A}$, where $\mathsf{Hdr}^* = x^*$.

2. For the first $q_{sk} - i - 1$ functional key queries, the adversary $\mathcal{A}'_{i\mathcal{O}}$ responds in the same manner as in game G_3. For the last i queries, $\mathcal{A}'_{i\mathcal{O}}$ responds in the same manner as in game G_4.
3. For the $(q_{sk} - i)$'th functional key query (i, f) from $\mathcal{A}$, the adversary $\mathcal{A}'_{i\mathcal{O}}$ first constructs two circuits $G[i, f, \mathsf{PRF}_{enc}]$ and $G'[i, f, \mathsf{PRF}_{enc}^{\overline{\{x^*,S^*\}}}, x^*, S^*, K_{S^*}, c^*]$, then delivers them to its challenger $\mathcal{B}$, and from which it gets an obfuscated circuit GG, where GG is either the obfuscation of the circuit G or the obfuscation of the circuit G'. Finally, the adversary $\mathcal{A}'_{i\mathcal{O}}$ sets $sk_f^i = GG$ and sends sk_f^i to $\mathcal{A}$.
4. Finally the adversary $\mathcal{A}$ outputs a bit b' which is defined as the output of the experiment.

Note that when G and G' are given inputs (S, x, c_m) such as $(x, S) \neq (x^*, S^*)$, both of them decrypt c_m under the same key K_S. This is because the key $K_S = \mathsf{PRF}_{enc}^{\overline{\{x^*,S^*\}}}(x, S)$ computed in program G' is equal to the key $K_S = \mathsf{PRF}_{enc}(x, S)$ computed in program G. When given an input (S, x, c_m) such that $(x, S) = (x^*, S^*)$, they also use the identical key $K_{S^*} = \mathsf{PRF}_{enc}(x^*, S^*)$ to compute their outputs. Thus, the indistinguishability of the two games may be reduced to the security of the $i\mathcal{O}$. By a hybrid argument, we have $\Pr[\mathsf{G}_3] - \Pr[\mathsf{G}_4] \leq q_{sk}.\mathsf{Adv}_{i\mathcal{O},\mathcal{A}'_{i\mathcal{O}}}^{i\mathcal{O}}$. □

Lemma 5. *Let* $\Pr[\mathsf{G}_4]$ *and* $\Pr[\mathsf{G}_5]$ *respectively denote the probability that an adversary* $\mathcal{A}$ *wins in game* G_4 *and game* G_5. *Then there exists an adversary* $\mathcal{A}_{prf}$ *that can break the security of the puncturable* PRF.

$$\Pr[\mathsf{G}_4] - \Pr[\mathsf{G}_5] \leq \mathsf{Adv}_{\mathsf{PRF},\mathcal{A}_{prf}}^{pr}. \tag{5}$$

Proof. Note that the difference between G_4 and G_5 is that in G_4, we compute the group key $K_{S^*} \leftarrow \mathsf{PRF}_{enc}(x^*, S^*)$, while in G_5, we compute $K_{S^*} \leftarrow_\$ \mathcal{K}$. In order to prove G_4 and G_5 to be computationally indistinguishable, we assume that there exists an adversary $\mathcal{A}$ that can distinguish G_4 and G_5, then we can use $\mathcal{A}$ to construct an adversary $\mathcal{A}_{prf}$ that can break the security of the puncturable PRF. The adversary $\mathcal{A}_{prf}$ is constructed as follows.

1. The adversary $\mathcal{A}_{prf}$ first constructs challenge ciphertext as follows. For the challenge query (S^*, m_0, m_1), the adversary $\mathcal{A}_{prf}$ first samples $x^* \leftarrow_\$ \{0, 1\}^{2\lambda}$, then delivers (x^*, S^*) to its challenger $\mathcal{B}$, and from which it gets $(K_{S^*}, \mathsf{PRF}_{enc}^{\overline{\{x^*,S^*\}}})$, where $K_{S^*} \leftarrow \mathsf{PRF}_{enc}(x^*, S^*)$ or $K_{S^*} \leftarrow_\$ \mathcal{K}$. Next, it computes $c^* = K_{S^*} \oplus m_0$ and sets the master public key $mpk = i\mathcal{O}(P'_{\mathsf{FBRB}}[\mathsf{PRF}_{enc}^{\overline{\{x^*,S^*\}}}])$. Finally, it sends the challenge ciphertext $ct^* = (S^*, \mathsf{Hdr}^*, c^*)$ and the master public key mpk to $\mathcal{A}$, where $\mathsf{Hdr}^* = x^*$.
2. For every functional key query (i, f) from $\mathcal{A}$, $\mathcal{A}_{prf}$ first constructs $G'[i, f, \mathsf{PRF}_{enc}^{\overline{\{x^*,S^*\}}}, x^*, S^*, K_{S^*}, c^*]$, and then uses it to compute $sk_f^i \leftarrow i\mathcal{O}(G')$, and finally it sends sk_f^i to the adversary $\mathcal{A}$.
3. Finally the adversary $\mathcal{A}$ outputs a bit b' which is defined as the output of the experiment.

Obviously, the adversary $\mathcal{A}_{prf}$ can perfectly simulate the environment for $\mathcal{A}$. Namely, when $K_{S^*} = \mathsf{PRF}_{enc}(x^*, S^*)$, it simulates G_4 for $\mathcal{A}$; when $K_{S^*} \leftarrow_\$ \mathcal{K}$, it simulates G_5 for $\mathcal{A}$. Thus we have $\Pr[\mathsf{G}_4] - \Pr[\mathsf{G}_5] \leq \mathsf{Adv}_{\mathsf{PRF},\mathcal{A}_{prf}}^{pr}$. □

Lemma 6. *Let* $\Pr[\mathsf{G}_5]$ *and* $\Pr[\mathsf{G}_6]$ *respectively denote the probability that an adversary* $\mathcal{A}$ *wins in game* G_5 *and game* G_6. *Then we have*

$$\Pr[\mathsf{G}_5] = \Pr[\mathsf{G}_6]. \tag{6}$$

Proof. In game G_5, since $c^* = K_{S^*} \oplus m_0$ is computed under a uniform and random group key K_{S^*}, c^* is also uniform and random, while in game G_6, $c^* = K_{S^*} \oplus m_1$ is also uniform and random due to the uniform and random K_{S^*}. Thus, G_5 and G_6 are identical. □

Lemma 7. *Let* $\Pr[\mathsf{G}_6]$ *and* $\Pr[\mathsf{G}_7]$ *respectively denote the probability that an adversary* $\mathcal{A}$ *wins in game* G_6 *and game* G_7. *Then there exists an adversary* $\mathcal{A}_{prf}$ *that can break the security of the puncturable* PRF.

$$\Pr[\mathsf{G}_6] - \Pr[\mathsf{G}_7] \leq \mathsf{Adv}_{\mathsf{PRF},\mathcal{A}_{prf}}^{pr}. \tag{7}$$

Proof. The only difference in game G_6 and game G_7 is that in game G_7, the ciphertext $c^* = K_{S^*} \oplus m_1$ is computed under the key $K_{S^*} \leftarrow \mathsf{PRF}_{enc}(x^*, S^*)$, while in game G_6, c^* is computed under a random key $K_{S^*} \leftarrow_\$ \mathcal{K}$. Subsequently, we will prove that the indistinguishability of the two games can be reduced to the security of the puncturable PRF. Assume that there exists an adversary $\mathcal{A}$ that can distinguish G_6 and G_7, then we can construct an adversary $\mathcal{A}_{prf}$ that can use $\mathcal{A}$ to break the security of the puncturable PRF. Let $\mathcal{B}$ be the PRF challenger. Then we construct the adversary $\mathcal{A}_{prf}$ as follows.

1. On receive the challenge query (S^*, m_0, m_1) submitted by the adversary $\mathcal{A}$, the adversary $\mathcal{A}_{prf}$ first samples $x^* \leftarrow_\$ \{0, 1\}^{2\lambda}$ and then delivers (x^*, S^*) to its challenger $\mathcal{B}$, and from which it obtains K_{S^*} and $\mathsf{PRF}_{enc}^{\overline{\{x^*,S^*\}}}$, where K_{S^*} is either computed as $K_{S^*} \leftarrow \mathsf{PRF}_{enc}(x^*, S^*)$ or $K_{S^*} \leftarrow_\$ \mathcal{K}$. Next, it computes $c^* = K_{S^*} \oplus m_1$ and $mpk = i\mathcal{O}(P'_{\mathsf{FBRB}}[\mathsf{PRF}_{enc}^{\overline{\{x^*,S^*\}}}])$. Finally, it sends the challenge ciphertext (S^*, Hdr, c^*) and the master public key mpk to $\mathcal{A}$, where $\mathsf{Hdr}^* = x^*$.
2. For each functional key query (i, f) from the adversary $\mathcal{A}$, the adversary $\mathcal{A}_{prf}$ first constructs the circuit $G'[i, f, \mathsf{PRF}_{enc}^{\overline{\{x^*,S^*\}}}, x^*, S^*, K_{S^*}, c^*]$, then sets the functional key as $sk_f^i \leftarrow i\mathcal{O}(G')$ and sends sk_f^i to $\mathcal{A}$.
3. Finally the adversary $\mathcal{A}$ outputs a bit b' which is defined as the output of the experiment.

Obviously, when $K_{S^*} \leftarrow_{\$} \mathcal{K}$, the adversary $\mathcal{A}_{prf}$ simulates G_6 for $\mathcal{A}$, when $K_{S^*} \leftarrow \mathsf{PRF}_{enc}(x^*, S^*)$, it simulates G_7 for $\mathcal{A}$. Therefore, the indistinguishability of the two games can be reduced to the security of the puncture PRF. Thus we have $\Pr[\mathsf{G}_6] - \Pr[\mathsf{G}_7] \le \mathsf{Adv}^{pr}_{\mathsf{PRF},\mathcal{A}_{prf}}$. □

Lemma 8. *Let* $\Pr[\mathsf{G}_7]$ *and* $\Pr[\mathsf{G}_8]$ *respectively denote the probability that an adversary* $\mathcal{A}$ *wins in* G_7 *and* G_8. *Then for any polynomial* q_{sk}, *there exists an adversary* $\mathcal{A}'_{i\mathcal{O}}$ *which can break the security of the* $i\mathcal{O}$.

$$\Pr[\mathsf{G}_7] - \Pr[\mathsf{G}_8] \le q_{sk}.\mathsf{Adv}^{i\mathcal{O}}_{i\mathcal{O},\mathcal{A}'_{i\mathcal{O}}}. \tag{8}$$

Where q_{sk} *denotes the maximal number that the adversary is allowed to make functional key queries.*

Proof. The same as Claim 4. □

Lemma 9. *Let* $\Pr[\mathsf{G}_8]$ *and* $\Pr[\mathsf{G}_9]$ *respectively denote the probability that an adversary* $\mathcal{A}$ *wins in game* G_8 *and game* G_9. *Then there exists an adversary* $\mathcal{A}_{prg}$ *which breaks the security of the* PRG.

$$\Pr[\mathsf{G}_8] - \Pr[\mathsf{G}_9] \le \mathsf{Adv}^{pr}_{\mathsf{PRG},\mathcal{A}_{prg}}. \tag{9}$$

Proof. The same as Claim 2 □

Lemma 10. *Let* $\Pr[\mathsf{G}_9]$ *and* $\Pr[\mathsf{G}_{10}]$ *respectively denote the probability that an adversary* $\mathcal{A}$ *wins in game* G_9 *and game* G_{10}. *Then there exists an adversary* $\mathcal{A}_{i\mathcal{O}}$ *which breaks the security of the* $i\mathcal{O}$.

$$\Pr[\mathsf{G}_9] - \Pr[\mathsf{G}_{10}] \le \mathsf{Adv}^{i\mathcal{O}}_{i\mathcal{O},\mathcal{A}_{i\mathcal{O}}}. \tag{10}$$

Proof. The same as 3. □

By Lemmas 1–10 and the assumptions we make, Theorem 1 holds.

Remark 3. In [2], Ananth et al. present a general transformation from selective security to adaptive security for functional encryption. By applying this transformation to our proposed scheme, we get the adaptive version.

In the IND-CPA security game described above, if we allow the adversary to make decryption queries, then the security becomes IND-CCA. Similar to IND-CPA, IND-CCA is also classified into selective and adaptive versions. Here we first give the selective IND-CCA definition by the experiment $\mathsf{Exp}^{sel-cca}(b)$ which is parameterized by the total number of parties N and an adversary $\mathcal{A}$:

- Setup. Before the master public key is generated, the adversary first submits to a challenge set S^* and two challenge messages m_0 and m_1 to the challenger. Then, the challenger $\mathcal{C}$ runs $\mathsf{FBRB.Setup}(\lambda, N)$ to generate the master public key and master secret key (mpk, msk).
- Challenge. For a challenge bit $b \in \{0, 1\}$, the challenger first runs $\mathsf{FBRB.Enc}(mpk, S^*, m_b)$ to generate $(\mathsf{Hdr}^*, K_{S^*})$. Then it sends the challenge ciphertext $ct^* = (S^*, \mathsf{Hdr}^*, c^*)$ and the master public key mpk to the adversary $\mathcal{A}$, where c^* is the encryption of m_b under the key K_{S^*}.
- Functional Key Query. The adversary adaptively queries the challenger with its choice of identity $i \in [N]$ and function $f \in \mathcal{F}$ such that $f(m_0) = f(m_1)$. For each such query, the challenger replies with $sk^i_f \leftarrow \mathsf{FBRB.KeyGen}(msk, i, f)$.
- Decryption Query. For each query (i, f, S, x, c_m) from $\mathcal{A}$ such that $(S, x) \neq (S^*, x^*)$, the challenger first generates $sk^i_f \leftarrow \mathsf{BFE.KeyGen}(msk, i, f)$ and then $f(m) \leftarrow \mathsf{FBRB.Dec}(sk^i_f, S, x, c_m)$, and finally it sends $f(m)$ to $\mathcal{A}$.
- Output. The adversary outputs a bit b' which is defined as the output of the experiment.

We define the advantage that an adversary $\mathcal{A}$ wins in the above experiment $\mathsf{Exp}^{sel-cca}(b)$ as $\mathsf{Adv}^{sel-cca}_{\mathsf{FBRB},\mathcal{A}}(\lambda) = \Pr[\mathsf{Exp}^{sel-cca}(0) = 1] - \Pr[\mathsf{Exp}^{sel-cca}(1) = 1]$.

Definition 5 (Selective IND-CCA Security)**.** A scheme FBRB is **selective** IND-CCA secure if for any polynomial N and any PPT adversaries $\mathcal{A}$, the advantage function $\mathsf{Adv}^{sel-cca}_{\mathsf{FBRB},\mathcal{A}}(\lambda)$ defined above is negligible in the security parameter λ.

Unlike the selective IND-CCA security, in the adaptive version, we do not require the adversary to submit challenge set and challenge messages in advance. We denote the adaptive IND-CCA security by the experiment $\mathsf{Exp}^{adp-cca}(b)$ and the advantage that the adversary wins in this experiments is defined as $\mathsf{Adv}^{adp-cca}_{\mathsf{FBRB},\mathcal{A}}(\lambda) = \Pr[\mathsf{Exp}^{adp-cca}(0) = 1] - \Pr[\mathsf{Exp}^{adp-cca}(1) = 1]$.

Definition 6 (Adaptive IND-CCA Security)**.** A scheme is **adaptive** IND-CCA secure if for any polynomial N and any PPT adversary $\mathcal{A}$, the advantage function $\mathsf{Adv}^{adp-cca}_{\mathsf{FBRB},\mathcal{A}}(\lambda)$ defined above is negligible in the security parameter λ.

Theorem 2. *If* PRG *is a secure pseudorandom generator,* PRF_{enc} *is a secure puncturable pseudorandom function,* $i\mathcal{O}$ *is a secure indistinguishable obfuscator, then the scheme* FBRB = (FBRB.Setup, FBRB.KeyGen, FBRB.Enc, FBRB.Dec) *constructed in Section 6 is a selective IND-CCA secure over the function space* $\mathcal{F}$ *and message space* $\mathcal{M}$.

Proof. Since the IND-CCA security is similar to the IND-CPA security except that it additionally allows the adversary to make decryption queries. Therefore, we prove Theorem 2 by adding decryption queries to every reduction. For simplicity, we now

Table 1
Notations and their meanings used in Table 2.

Symbol	Corresponding meanings
$\lvert e_1\rvert$	the bit-length of a traditional PKE ciphertext
$\lvert e_2\rvert$	the bit-length of a traditional PKE ciphertext
$\lvert\pi\rvert$	the bit-length of the NIZK proof π that proves ciphertexts e_1 and e_2 that encrypt the same message
$\lvert S\rvert$	the size of a set S
$\lvert m\rvert$	the bit-length of a message m

Table 2
Storage overhead on cloud server.

	Storage overhead on cloud server
[15]	$2\lvert e_1\rvert + \lvert e_2 + \lvert\pi\rvert$
[10]	$log\lvert S\rvert + 2\lambda + \lvert m\rvert$
FBRB	$log\lvert S\rvert + 2\lambda + \lvert m\rvert$

only illustrate how to embed decryption queries into the proofs of Lemmas 1–10. By observation, Lemmas 1 and 6 satisfy the IND-CCA security directly, and in Lemmas 2, 3, and 4, the reductions can also simulate the decryption queries, since the adversaries $\mathcal{A}_{prg}$, $\mathcal{A}_{i\mathcal{O}}$ and $\mathcal{A}'_{i\mathcal{O}}$ possess PRF_{enc}. But, in Lemmas 5 and 7, since the adversary $\mathcal{A}_{prf}$ has no PRF_{enc}, the reduction can not complete the simulations directly. Fortunately, $\mathcal{A}_{prf}$ can queries its challenger to get the punctured key $\mathsf{PRF}_{enc}^{\overline{\{x^*,S^*\}}}$, and under this key the decryption queries can be perfectly simulated. □

Remark 4. For now, we don't know how to directly transform the selective IND-CCA security to the adaptive IND-CCA security for functional broadcast encryption, thus at this point, our scheme only achieve selective security. But for applications in this paper, the selective IND-CCA security is enough.

Remark 5. Particularly, note that the scheme described above implies the secret-key version, which can be achieved by simply replacing x in the encryption algorithm with a random string and taking the master secret-key PRF_{enc} as an input of the encryption algorithm which can be directly used to compute the group key K_S, and the security proof is similar to the one demonstrated in 7.

8. Performance analysis

We analyze the performance of FBRB and provide a comparison between FBRB and existing schemes [15], [10] due to the similar properties of those schemes in [10,15] with ours. All the experiments are executed on a laptop with Windows 10 system with Intel 2 Cores 8 i78565 CPU, 1.80 GHz and 1.99 GHz and 8.0GB DDR of RAM. We use C programming language. We select 80 bits security parameter for analysis.

8.1. Storage costs

We first analyze the storage costs of our scheme, and then give a comparison of our scheme with the schemes in [10,15]. The total storage costs on the cloud server of our scheme is $log\lvert S\rvert + 2\lambda + \lvert m\rvert$, where $\lvert S\rvert$ denotes the size of an authorized set S that a data sender specifies, λ denotes a security parameter, and $\lvert m\rvert$ denotes the bit-length of a message m. The storage costs on the cloud server of the scheme in [15] is $\lvert e_1\rvert + \lvert e_2\rvert + \lvert\pi\rvert$, where the $\lvert e_1\rvert$ and $\lvert e_2\rvert$ denote the bit-length of ciphertexts e_1 and e_2 of a traditional public-key encryption (PKE), and $\lvert\pi\rvert$ denotes the bit-length of a NIZK proof π.

We give the storage cost comparison between our scheme and the schemes in [10,15]. First, we define some notations in Table 1 to represent the storage costs. The comparison results are demonstrated in Table 2. In Fig. 7, we show how the storage cost changes with the size of the authorized subset that a data sender specifies. We assume $2\lambda + \lvert m\rvert = c$ and $2\lvert e_1\rvert + \lvert e_2\rvert + \lvert\pi\rvert = kc$, where $k \in \mathbb{N}$. As Fig. 7 shows, if the size of the authorized subset is 32, then it will only increase by one bit of ciphertext than that induced by the size 16. The results demonstrate that the larger the size of the authorized subset, the smaller the growth span of the ciphertext. In reality, especially in a eHealth system, up to 32 doctors participating in a patient's diagnosis is enough. Actually, the storage costs of [10] are the same as that of FBRB and the storage costs of [15] are kc. Compared with [15], FBRB is more efficient in terms of storage overhead.

8.2. Computational costs

For the computational costs, here, we only consider the computational costs on the data sender and data receiver sides. In FBRB, the data sender's computational costs is that of performing an encryption operation, which needs a PRG operation and an execution of an obfuscated circuit on a polynomial-length input. On the data receiver side, the computational costs

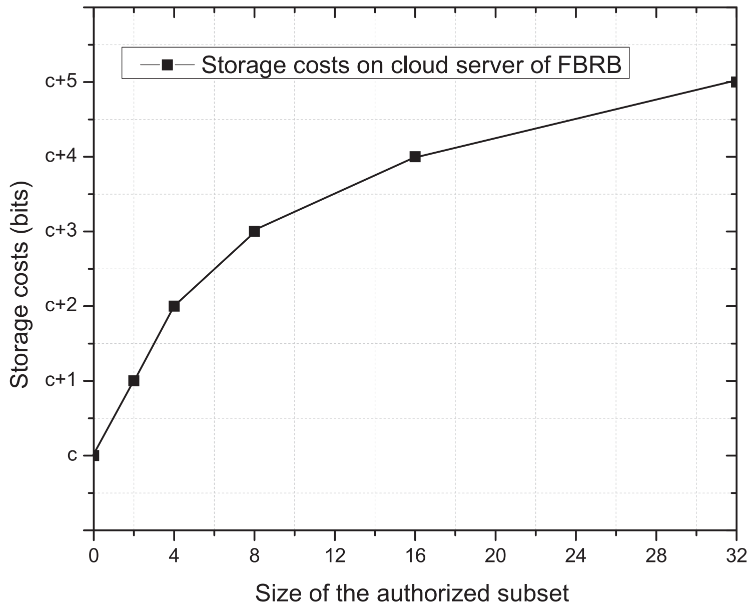

Fig. 7. System model.

Table 3
Notations and their meanings used in 4.

Symbol	Corresponding meanings
2(*PKE*)	run 2 times of traditional PKE encryption algorithms
1(*Proof*)	run 1 time of NIZK proof algorithm that proves two ciphertexts e_1 and e_2 that encrypt the same message
1(*PRG*)	run 1 time of PRG algorithm
1($i\mathcal{O}$)	run 1 time of obfuscated circuit on a polynomial-length input

Table 4
Computational costs on data sender and data receiver.

	Costs on data sender	Costs on data receiver
[15]	$2(PKE)+1(Proof)$	$1(i\mathcal{O})$
[10]	$1(PRG)+1(i\mathcal{O})$	$1(i\mathcal{O})$
FBRB	$1(PRG)+1(i\mathcal{O})$	$1(i\mathcal{O})$

Table 5
Functionality comparison.

	file-based PEAC?	receiver-based PEAC?
[15]	Yes	No
[10]	No	Yes
FBRB	Yes	Yes

are only an execution of an obfuscated circuit on a polynomial-length input. The comparison results between FBRB and that in [10,15] are shown in Table 4, where the notations and their meanings used in Table 4 is demonstrated in Table 3.

8.3. Functionality comparisons

In this section, we make a comparison between the functionalities implemented in FBRB and those in [10,15] in Table 5, where the notations “file-based PEAC ?” and “receiver-based PEAC ?” denote whether the file-based PEAC and receiver-based PEAC are implemented in a scheme, respectively. In addition, the notation “Yes” indicates that the file-based PEAC or the receiver-based PEAC is implemented in a scheme, and “No” indicates that the file-based PEAC or the receiver-based PEAC is not implemented in a scheme. The comparison results between FBRB and [10,15] demonstrate that the obvious advantage achieved in FBRB.

9. Further discussion

Compared with previous data sharing schemes, where only either the file-based PEAC or receiver-based PEAC is implemented, while in FBRB, FBRB provides the file-based PEAC and receiver-based PEAC for data sharing system without reprocessing the cloud server-stored data, FBRB not only enables the data sender to control which receiver is allowed to receive the data sender's data, but also has the ability to control which part of the data is allowed to share by these receivers. Furthermore, since FBRB does not need preprocessing the cloud server-stored data, it can be easily ported to other systems that require the same functionality.

Above analysis shows that FBRB achieves the high storage performance on the cloud server side. But, we have to accept the fact that both the data sender and data receiver need to compute the obfuscated program. Note that the obfuscated program in FBRB is determined by the security parameter and the PRF key, and would not be changed along with the change of the inputs of the obfuscated program. Therefore, it is only one-time computation to generate the obfuscated program. Although the current obfuscation candidate [15,17,24]) are not very efficient in practical application, due to their unrealistic polynomial-time constructions, as analyzed in previous works [26,35], it is reasonable to believe that $i\mathcal{O}$ with reasonable performance will be realized in the near future.

10. Conclusions and future works

In this paper, we have proposed a primitive of functional broadcast encryption (FBE) that achieves both the file-based PEAC and receiver-based PEAC. FBE enables a data sender to share a set of files with a group of receivers, where the data sender can control which part of the data set can be accessed by whom. We also have presented a concrete construction of FBE called FBRB. We have formally prove the security of FBRB and gave a comprehensive performance analysis to show the efficiency of FBRB.

Regarding to the future work, we will first consider how to construct an FBE scheme on other cryptographic primitives rather than $i\mathcal{O}$ to improve the efficiency. Then we will investigate how to design an FBE scheme under some standard assumptions (e.g., DDH assumption).

Declaration of conflict of interest statement

All authors of this manuscript have directly participated in planning, execution, and/or analysis of this study (if not, specify). The contents of this manuscript have not been copyrighted or published previously. The contents of this manuscript are not now under consideration for publication elsewhere. The contents of this manuscript will not be copyrighted, submitted, or published elsewhere while acceptance by Urologic. Oncology: Seminars and Original Investigations is under consideration There are no directly related manuscripts or abstracts, published or unpublished, by any authors of this manuscript. No financial support or incentive has been provided for this manuscript(if financial support was provided, please specify below). I am sole author of this manuscript. I am one author signing on behalf of all co-authors of this manuscript, and attesting to the above.

Acknowledgements

The first author is supported by the National Key Research and Development Program of China (Grant No. 2017YFB0802000) and the National Natural Science Foundation of China (Grant Nos. NSFC61702007, NSFC61572318) and Other Foundations (Grant Nos. gxbjZD27, KJ2018A0533, XWWD201801, ahnis20178002, KJ2017A519, 16ZB0140). The third author is supported by the National Key Research and Development Program of China (Grant No. 2017YFB0802000) and the National Natural Science Foundation of China (Grant No. U1705264). The fifth author is supported in part by National Key Research and Development Program of China (Grant No. 2017YFB0802000), the National Natural Science Foundation of China (Grant Nos. 61877011, 61472084, U1536205), Shanghai Innovation Action Project (Grant No. 16DZ1100200), Shanghai Science and Technology Development Funds(Grant No. 16JC1400801), and Shandong Provincial Key Research and Development Program of China (Grant Nos. 2017CXG0701, 2018CXGC0701). The sixth author is supported by the National Natural Science Foundation of China (Grant Nos. 61822202, 61872089, 61872087).

References

[1] M. Abdalla, R. Gay, M. Raykova, H. Wee, Multi-input inner-product functional encryption from pairings, in: Proceedings of the EUROCRYPT, Springer, 2017, pp. 601–626.
[2] P. Ananth, Z. Brakerski, G. Segev, V. Vaikuntanathan, From selective to adaptive security in functional encryption, in: Proceedings of the CRYPTO, Springer, 2015, pp. 657–677.
[3] P. Ananth, A. Jain, Indistinguishability obfuscation from compact functional encryption, in: Proceedings of the CRYPTO, Springer, 2015, pp. 308–326.
[4] M. Bellare, P. Rogaway, The security of triple encryption and a framework for code-based game-playing proofs, in: Proceedings of the EUROCRYPT, Springer, 2006, pp. 409–426.
[5] D. Boneh, C. Gentry, B. Waters, Collusion resistant broadcast encryption with short ciphertexts and private keys, in: Proceedings of the CRYPTO, Springer, 2005, pp. 258–275.
[6] D. Boneh, A. Sahai, B. Waters, Functional encryption: definitions and challenges, in: Proceedings of the TCC, Springer, 2011, pp. 253–273.
[7] D. Boneh, B. Waters, A fully collusion resistant broadcast, trace, and revoke system, in: Proceedings of the CCS, Springer, 2006, pp. 211–220.

[8] D. Boneh, B. Waters, Constrained pseudorandom functions and their applications, in: Proceedings of the ASIACRYPT, Springer, 2013, pp. 280–300.
[9] D. Boneh, B. Waters, Conjunctive, subset, and range queries on encrypted data, in: Proceedings of the TCC, Springer, 2017, pp. 535–554.
[10] D. Boneh, M. Zhandry, Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation, Algorithmica 79 (4) (2017) 1233–1285.
[11] E. Boyle, S. Goldwasser, I. Ivan, Functional signatures and pseudorandom functions, in: Proceedings of the PKC, Springer, 2014, pp. 501–519.
[12] X. Fan, Q. Tang, Making public key functional encryption function private, distributively, in: Proceedings of the PKC, Springer, 2018, pp. 218–244.
[13] A. Fiat, M. Naor, Broadcast encryption, in: CRYPTO, Springer, 1993, pp. 480–491.
[14] J.A. Garay, J. Staddon, A. Wool, Long-lived broadcast encryption, in: Proceedings of the CRYPTO, Springer, 2000, pp. 333–352.
[15] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, B. Waters, Candidate indistinguishability obfuscation and functional encryption for all circuits, SIAM J. Comput. 45 (3) (2016) 882–929.
[16] R. Gay, L. Kowalczyk, H. Wee, Tight adaptively secure broadcast encryption with short ciphertexts and keys, SCN (2018) 123–139.
[17] C. Gentry, A.B. Lewko, A. Sahai, B. Waters, Indistinguishability obfuscation from the multilinear subgroup elimination assumption, in: Proceedings of the FOCS, IEEE, 2015, pp. 151–170.
[18] C. Gentry, B. Waters, Adaptive security in broadcast encryption systems (with short ciphertexts), in: Proceedings of the EUROCRYPT, Springer, 2009, pp. 171–188.
[19] O. Goldreich, S. Goldwasser, S. Micali, How to construct Randolli functions, in: Proceedings of the FOCS, IEEE, 1984, pp. 464–479.
[20] V. Goyal, A. Jain, V. Koppula, A. Sahai, Functional encryption for randomized functionalities, in: Proceedings of the TCC, 2005, pp. 325–351.
[21] J. Katz, A. Sahai, B. Waters, Predicate encryption supporting disjunctions, polynomial equations, and inner products, in: Proceedings of the EUROCRYPT, Springer, 2008, pp. 146–162.
[22] A. Kiayias, S. Papadopoulos, N. Triandopoulos, T. Zacharias, Delegatable pseudorandom functions and applications, in: Proceedings of the CCS, ACM, 2013, pp. 669–684.
[23] J. Li, Y.Y. Huang, Y. Wei, S.Y. Lv, Z.L. Liu, C.Y. Dong, W.J. Lou, Searchable symmetric encryption with forward search privacy, IEEE Trans. Dependable Secure Comput. (99) (2019) 1.
[24] R. Pass, K. Seth, S. Telang, Indistinguishability obfuscation from semantically-secure multilinear encodings, in: Proceedings of the CRYPTO, Springer, 2014, pp. 500–517.
[25] Z.L. Liu, B. Li, Y.Y. Huang, J. Li, Y. Xiang, W. Pedrycz., Newmcos: towards a practical multi-cloud oblivious storage scheme, IEEE Trans. Knowl. Data Eng. (99) (2019) 1.
[26] K. Ramchen, B. Waters, Fully secure and fast signing from obfuscation, in: Proceedings of the CCS, ACM, 2014, pp. 659–673.
[27] T. Ristenpart, H. Shacham, T. Shrimpton, Careful with composition: Limitations of the indifferentiability framework, in: Proceedings of the EUROCRYPT, Springer, 2011, pp. 487–506.
[28] A. Sahai, B. Waters, How to use indistinguishability obfuscation: deniable encryption, and more, in: Proceedings of the STOC, ACM, 2014, pp. 475–484.
[29] H. Wang, K. Chen, J.K. Liu, Z. Hu, Y. Long, Access control encryption with efficient verifiable sanitized decryption, Inf. Sci. 465 (2018) 72–85.
[30] H. Wang, K. Chen, B. Qin, Z. Hu, LR-RRA-CCA secure functional encryption for randomized functionalities from trapdoor HPS and LAF, Sci. China Inf. Sci. 61 (5) (2018) 058101:1–058101:3.
[31] H. Wang, K. Chen, B. Qin, X. Lai, Y. Wen, A new construction on randomized message-locked encryption in the standard model via UCEs, Sci. China Inf. Sci. 60 (5) (2017) 052101:1–052101:19.
[32] S. Xu, G. Yang, Y. Mu, R.H. Deng, Secure fine-grained access control and data sharing for dynamic groups in the cloud, IEEE Trans. Inf. Forens. Secur. 13 (8) (2018) 2101–2113.
[33] X. Zhang, H. Wang, C. Xu, Identity-based key-exposure resilient cloud storage public auditing scheme from lattices, Inf. Sci. 472 (2019) 223–234.
[34] Y. Zhang, C. Xu, H. Li, K. Yang, J. Zhou, X. Lin, HealthDep: An efficient and secure deduplication scheme for cloud-assisted ehealth systems, IEEE Trans. Inf. Forens. Secur. 14 (9) (2018) 4101–4112.
[35] Y. Zhang, C. Xu, X. Liang, H. Li, Y. Mu, X. Zhang, Efficient public verification of data integrity for cloud storage systems from indistinguishability obfuscation, IEEE Trans. Inf. Forens. Secur. 12 (3) (2017) 676–688.
[36] Y. Zhang, C. Xu, X. Lin, X. Shen, Blockchain-based public integrity verification for cloud storage against procrastinating auditors, IEEE Trans. Cloud Comput. (2019) to appear, doi:10.1109/TCC.2019.2908400.
[37] L. Zhou, C. Su, X. Sun, X. Zhao, K.-K. R. Choo, Stag hunt and trust emergence in social networks, Future Gener. Comput. Syst. 88 (2018) 168–172.
[38] C. Zuo, J. Shao, J. Liu, G. Wei, Y. Ling, Fine-grained two-factor protection mechanism for data sharing in cloud storage, IEEE Trans. Inf. Forens. Secur. (2018) 186–196.

Huige Wang received the Ph.D. degree from the Shanghai Jiaotong University, China. She is currently an Associate Professor with the Department of Computer, Anhui Science and Technology University, China. She is currently doing a Post-doc in the School of Computer Science, Fudan University. Her research interests are information security and cryptography, etc.

Yuan Zhang received the B.S. degree in University of Electronic Science Technology of China (UESTC) in 2013, Chengdu, China. He is currently a Ph.D candidate in School of Computer Science and Engineering at University of Electronic Science Technology of China, and is a visiting Ph.D student in BBCR Lab, Department of ECE, University of Waterloo, Canada. His research interests are applied cryptography, data security, and blockchain technology. He is a student member of IEEE.

Kefei Chen received the B.S. and M.S. degrees in applied mathematics from Xidian University, Xian, in 1982 and 1985, respectively, and the Ph.D. degree from Justus-Liebig University Giessen, Germany in 1994. He is currently a professor in the School of Science, Hangzhou Normal University, from 1996-2012 he was a professor in the Department of Computer Science and Engineering, Shanghai Jiaotong University. His main research areas are cryptography, theory and technology of network security, etc.

Guangye Sui received the Ph.D. degree in Laval University in 2015, Canada. He is currently doing a Post-doc in the School of Computer Science at Fudan University, China. His research interests are cryptography, formal methods and blockchain technology.

Yunlei Zhao received the Ph.D degree from Fudan University, Shanghai, China. He is currently a professor with School of Computer Science, Fudan university, China. His research interest lies in theory and applications of cryptography.

Xinyi Huang received the Ph.D. degree from the University of Wollongong, Australia. He is currently a Professor with the School of Mathematics and Computer Science, Fujian Normal University, China, and a Co-Director of the Fujian Provincial Key Laboratory of Network Security and Cryptology. He has authored over 100 papers in refereed international conferences and journals. His research interests include applied cryptography and network security.