# Efficient File Sharing in Electronic Health Records*

Clémentine Gritti, Willy Susilo, Thomas Plantard

Centre for Computer and Information Security Research
School of Computer Science and Software Engineering
University of Wollongong, Australia
cjpg967@uowmail.edu.au, {wsusilo,thomaspl}@uow.edu.au

**Abstract.** The issue of handling electronic health records have become paramount interest to the practitioners and security community, due to their sensitivity. In this paper, we propose a framework that enables medical practitioners to securely communicate among themselves to discuss health matters, and the patients can be rest assured that the information will only be made available to eligible medical practitioners. Specifically, we construct a new cryptographic primitive to enable File Sharing in Electronic Health Records (FSEHR). This primitive enables doctors to read the information sent by the hospital, or by any other individuals (such as patients' health records), when the doctors have their 'license' validated by that given hospital. We construct such a cryptographic primitive and capture its security requirements in a set of security models. Subsequently, we present a concrete scheme, which is proven selectively chosen-ciphertext security (CCA-1) secure under the Decisional Bilinear Diffie-Hellman Exponent (DBDHE) assumption and fully collusion resistant.

**Keywords:** File Sharing, Electronic Health Records, Broadcast Encryption, Certificate-Based Encryption, Bilinear map, Chosen-Ciphertext Security.

## 1 Introduction

Electronic Health Records (EHR) have become of paramount interest to practitioners and security community, due to their data size as well as their security issues and sensitivity. In the existing literature, there have been a number of papers [9, 12, 13, 2] that discuss the way to secure this type of data. In this work, we take a new direction by proposing a framework that enables secure communication among medical practitioners. The communication channel enabled here will allow medical practitioners, i.e. medical doctors, to communicate among themselves as well as to review the patients' EHRs. Additionally, it also allows the hospital to broadcast any important information to the doctors who are working in that hospital, and this information will only be made available to

* This work is partially supported by the Australian Linkage Project LP120200052.

them. The issue is the fact that the doctors have their own rights in a hospital, delivered by the government or the medical legislators. Therefore, our framework should be able to specify which are the doctors that have access to the data by updating their license as authorized in that hospital. To generalize this scenario, we decouple several entities involved in this scenario, namely the hospital (as the data owner), the government and the other medical legislators, who grant the license for the doctors to work in that hospital. This scenario is depicted in Figure 1. We note that the hospital would be the entity that generates the encrypted data for the doctors, which includes all important information that the doctors need to know. Additionally, any other members (such as the nurse or any other entities outside the doctors) should also be able to send any information to the group of doctors in that hospital. This is to represent a case for instance when a patient provides his/her X-ray or blood test results to the hospital.

**Our Work.** Based on the above scenario, we present a new cryptographic notion called File Sharing in Electronic Health Records (FSEHR). In this system, there are four entities, namely the Group Manager (GM), a group of Certificate Authorities (CA), or simply of Certifiers, a set of doctors and the rest of the universe. The GM represents an entity such as the hospital in this scenario. A CA represents the government, the medical institute or other medical legislator who grants the rights to some doctors to work in a given hospital. The set of doctors is denoted as a set of users, $U_i$, who are the main players in this setting. The rest of the universe includes the patients, nurses, and any other players who are not captured in the above set of entities. The scenario of the FSEHR primitive is depicted in Figure 1. In this work, we provide a sound security model to capture this scenario.

One may think that FSEHR could be achieved easily by combining the two cryptographic primitives, namely certificate-based encryption and broadcast encryption schemes. Unfortunately this is false. The primary difficulty of achieving such a scheme is due to the need to achieve a shorter ciphertext, in comparison to merely combining the ciphertexts into one to achieve a linear size. Furthermore, the notion of certificate-based encryption usually allows a sender to interact with only a single receiver, with the help of a CA. Hence, it is clear that by a simple combination of the existing certificate-based encryption scheme and a broadcast encryption will lead a construction with a linear size of ciphertext (in number of both users and certifiers), which is undesirable (otherwise, it would be better for the broadcaster to just simply encrypt each ciphertext individually for each user). Moreover, we require that a FSEHR scheme to be fully collusion resistant, which follows the original notion of broadcast encryption schemes. Hence, even if all the users in the system collude, only the users in the selected subset with valid certificates can recover the plaintext.

**Related Work.** Modelling and securing authorization and access control to EHR has become a challenge. In [9], such a model, called Role-Based Access Control (RBAC), is proposed to inscrease the patient privacy and confidentiality of his data but it remains flexible to allow specific situations (for instance,

emergency cases). A literature review about security and privacy in EHR can be found in [2].

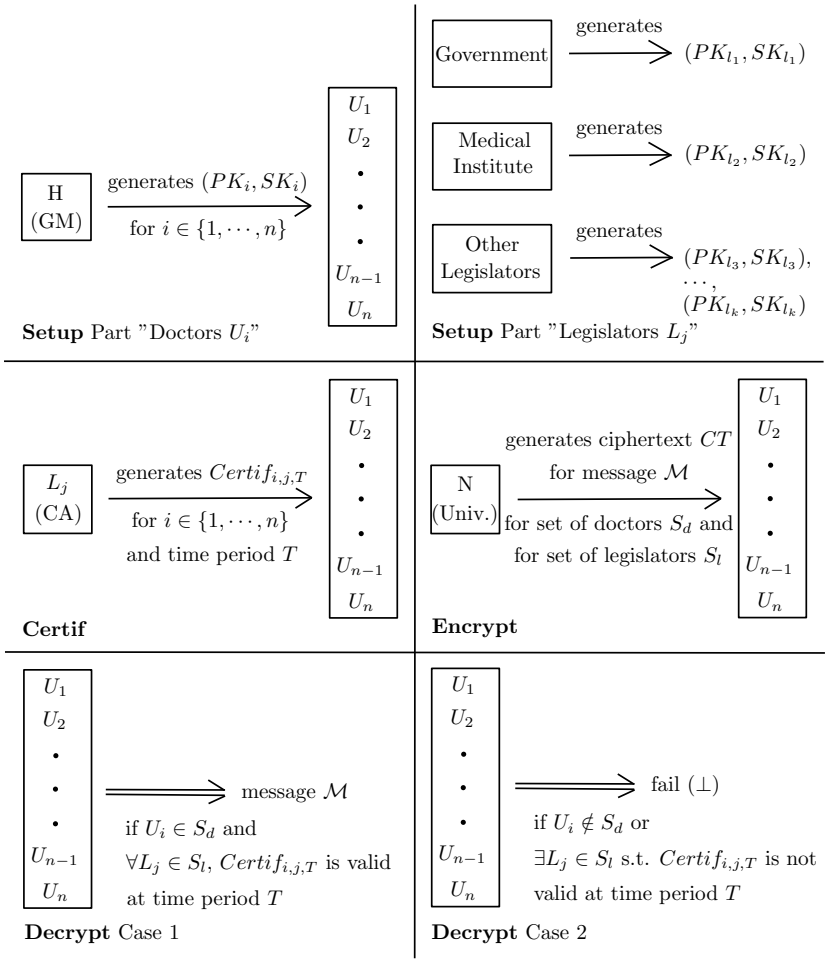


**Fig. 1.** A hospital $H$ (GM) generates and sends the public/secret key pairs $(PK_i, SK_i)$ for doctor $U_i$, where $i = 1, \cdots, n$. The government, the medical institute and the other medical legislators $L_j$ (CAs) create their public/secret key pairs $(PK_{l_1}, SK_{l_1}), (PK_{l_2}, SK_{l_2}), (PK_{l_3}, SK_{l_3}), \cdots, (PK_{l_k}, SK_{l_k})$ respectively. The legislators then compute and give the certificates $Certif_{i,j,l}$ for doctor $U_i$, where $i = 1, \cdots, n$, $j = 1, \cdots, k$, and time period $T$. Finally, a hospital staff member (Universe), for instance a nurse $N$, selects a subgroup of doctors $S_d = \{U_{i_1}, \cdots, U_{i_{|S_d|}}\}$ and a subgroup of legislators $S_l$, encrypts a message $\mathcal{M}$ for doctors in $S_d$, legislators in $S_l$ and time period $T$, and sends the resulting ciphertext $CT$. Only a doctor $U_{i_{i'}}$ in $S_d$, with certificate $Certif_{i,j,l}$ valid at time period $T$ and where $L_j \in S_l$, can recover $\mathcal{M}$.

Benaloh et al. [3] combined a Hierarchical Identity-Based Encryption (HIBE) and a searchable encryption to obtain a privacy-preserving and patient-centered EHR system. However, the patients have to create and manage manifold keys and check the credentials of the healthcare providers (doctors, nurses, ...). Narayan et al. [10] proposed an EHR system using a broadcast Ciphertext-Policy Attribute-Based Encryption (bCP-ABE) scheme, a variant of ABE system, and a Public key Encryption with Keyword Search (PEKS) scheme. Their system is secure, allows private search on health records by performing keyword matching without leaking information, and enable direct revocation of user access without having to re-encrypt the data. But the system is only designed for online access control. More recently, Akinyele et al. [1] presented a system along with a mobile app for iPhone for secure offline access to EHRs. They constructed their scheme based on ABE (Key-Policy and Ciphertext-Policy versions) and they developped a corresponding software library to help the implementation.

## 2 Preliminaries and Definitions

### 2.1 File Sharing in Electronic Health Records (FSEHR)

A File Sharing system in Electronic Health Records (FSEHR) comprises four algorithms ($\mathbf{Setup}, \mathbf{Certif}, \mathbf{Encrypt}, \mathbf{Decrypt}$):

1. $\mathbf{Setup}(\lambda, n, k)$ run by the group manager, takes as inputs the security parameter $\lambda$, the total number $n$ of users, and the total number $k$ of cerfitiers, output the public parameters $PK$, the public/secret key pair $(PK_i, SK_i)$ for user $i \in \{1, \cdots, n\}$. The public/secret key pair $(PK_{c_j}, SK_{c_j})$ are independently generated by the certifier $j \in \{1, \cdots, k\}$. We assume that $n$ is the bound of the universe, in order to pre-compute public/secret key pairs and allow users to join the system later.
2. $\mathbf{Certif}(PK, (PK_{c_j}, SK_{c_j}), PK_i, l)$ run by the certifier $j \in \{1, \cdots, k\}$, takes as inputs the public parameters $PK$, the certifier $j$'s public/secret key pair $(PK_{c_j}, SK_{c_j})$, the public key $PK_i$ for user $i \in \{1, \cdots, n\}$, and the time period $l$, output the certificate $Certif_{i,j,l}$ for $i$, $j$ and $l$.
3. $\mathbf{Encrypt}(PK, S_u, S_c, \{PK_i : i \in S_u\}, \{PK_{c_j} : j \in S_c\}, l)$, run by the sender belonging to the universe, takes as inputs the public parameters $PK$, the subset $S_u$ of users selected by the sender such that $S_u \subseteq \{1, \cdots, n\}$, the subset $S_c$ of certifiers selected by the sender such that $S_c \subseteq \{1, \cdots, k\}$, the public keys $PK_i$ for users $i \in S_u$, the certifier's public key $PK_{c_j}$ for certifers $j \in S_c$, and the time period $l$, output the header $Hdr$ and the session key $K$.
4. $\mathbf{Decrypt}(PK, S_u, S_c, l, (PK_i, SK_i), \{Certif_{i,j,l} : j \in S_c\}, Hdr)$ run by user $i$, takes as inputs the public parameters $PK$, the subset $S_u$ of users selected by the sender, the subset $S_c$ of certifiers selected by the sender, the time period $l$, the public/secret key pair $(PK_i, SK_i)$ and the certificates $Certif_{i,j,l}$ for user $i$ and certifier $j \in S_c$, and the header $Hdr$, output the session key $K$ if $i \in S_u$ and $Certif_{i,j,l}$ is valid for time period $l$; otherwise $\perp$.

We require that for user $i \in \{1,\cdots,n\}$ and certifier $j \in \{1,\cdots,k\}$ such that $(PK,(PK_i,SK_i),(PK_{c_j},SK_{c_j})) \leftarrow \mathbf{Setup}(\lambda,n,k)$, and subsets $S_u \in \{1,\cdots,n\}$ and $S_c \in \{1,\cdots,k\}$: if user $i \in S_u$, certifier $j \in S_c$ and $Certif_{i,j,l}$ is valid for time period $l$, then $K \leftarrow \mathbf{Decrypt}(PK,S_u,S_c,l,(PK_i,SK_i),\{Certif_{i,j,l} : j \in S_c\},\mathbf{Encrypt}(PK,S_u,S_c,\{PK_i : i \in S_u\},\{PK_{c_j} : j \in S_c\},l))$; otherwise $\perp \leftarrow \mathbf{Decrypt}(PK,S_u,S_c,l,(PK_i,SK_i),\{Certif_{i,j,l} : j \in S_c\},\mathbf{Encrypt}(PK,S_u,S_c,\{PK_i : i \in S_u\},\{PK_{c_j} : j \in S_c\},l))$.

We call the *header*, $Hdr$, as the encryption of the session key $K$. We call the *full header* as the header $Hdr$ along with the descriptions of the set $S_u$ of users and the set $S_c$ of certifiers selected by the sender. Without losing generalization, we only consider the header $Hdr$ when discussing the size of the scheme.

### 2.2 Security Requirements

We first present the definitions for chosen-plaintext attack (CPA) and chosen-ciphertext attack (CCA) securities. We adopt the security definitions from the certificate-based encryption [7]. There are two basic attacks which may be launced by an uncertified user or by the certifier. These are captured in the following two distinct games. In Game 1, the adversary plays the role of an uncertified user: it first proves that it knows the secret key of the uncertified user and then, it can make Decryption and Certification queries. In Game 2, the adversary plays the role of a trusted certifier: it first proves that it knows the secret key of the certifier and then, it can make Decryption queries. Eventually, we say that the FSEHR system is secure if no adversary can win either game.

**Game 1. Setup.** The challenger runs the algorithm $\mathbf{Setup}(\lambda,n,k)$ to obtain the public parameters $PK$, the public keys $PK_i$ for users $i \in \{1,\cdots,n\}$, and the public keys $PK_{c_j}$ for certifiers $j \in \{1,\cdots,k\}$, and gives them to $\mathcal{A}_1$.
**Certification Query Phase.** For time period $l$, for user $i \in \{1,\cdots,n\}$, and for $j \in \{1,\cdots,k\}$, the challenger first checks that $SK_i$ is the secret key corresponding to the public key $PK_i$. If so, it runs **Certif** and returns $Certif_{i,j,l}$ to the adversary $\mathcal{A}_1$; otherwise, it returns $\perp$.
**Decryption Query Phase.** For time period $l$, for user $i \in \{1,\cdots,n\}$, and for certifier $j \in \{1,\cdots,k\}$, the challenger first checks that $SK_i$ is the secret key corresponding to the public key $PK_i$. If so, it returns $\mathbf{Decrypt}(PK,S_u,S_c,l,(PK_i,SK_i),\{Certif_{i,j,l} : j \in S_c\},Hdr)$ to the adversary $\mathcal{A}_1$; otherwise, it returns $\perp$.
**Challenge.** For time period $l^*$, $\mathcal{A}_1$ outputs a challenge set $S_u^* \subseteq \{1,\cdots,n\}$. For user $i \in S_u^*$, the challenger first checks that $SK_i^*$ is the secret key corresponding to the public key $PK_i^*$. If so, it chooses a set $S_c^* \subseteq \{1,\cdots,k\}$ and a random bit $b \in_R \{0,1\}$, and runs $(Hdr^*,K^*) \leftarrow \mathbf{Encrypt}(PK^*,S_u^*,S_c^*,\{PK_i^* : i \in S_u^*\},\{PK_{c_j}^* : j \in S_c^*\},l^*)$. It then sets $K_b^* = K^*$, picks at random $K_{1-b}^*$ in the key space, and gives $(Hdr^*,K_b^*,K_{1-b}^*)$ to the adversary $\mathcal{A}_1$. Otherwise, it gives $\perp$.
**Guess.** The adversary $\mathcal{A}_1$ outputs its guess $b' \in_R \{0,1\}$ for $b$ and wins the game if $b = b'$, $(l^*,S_u^*,Hdr^*)$ was not the subject of a valid Decryption query after the Challenge, and $(l^*,S_u^*)$ was not subject of any valid Certification query.

We define $\mathcal{A}_1$'s advantage in attacking the File Sharing system in Electronic Health Records for Game 1 with parameters $(\lambda, n, k)$ as $Adv\mathsf{FSEHR}^{Game1}_{\mathcal{A}_1,n,k}(\lambda) = |Pr[b = b'] - \frac{1}{2}|$.

**Game 2. Setup.** The challenger runs the algorithm $\mathbf{Setup}(\lambda, n, k)$ to obtain the public parameters $PK$, the public keys $PK_i$ for users $i \in \{1, \cdots, n\}$, and the public keys $PK_{c_j}$ for certifiers $j \in \{1, \cdots, k\}$, and gives them to $\mathcal{A}_2$.
**Decryption Query Phase.** For time period $l$ and for certifier $j \in \{1, \cdots, k\}$, the challenger first checks that $SK_{c_j}$ is the secret key corresponding to the public key $PK_{c_j}$. If so, it returns $\mathbf{Decrypt}(PK, S_u, S_c, l, (PK_i, SK_i), \{Certif_{i,j,l} : j \in S_c\}, Hdr)$ to the adversary $\mathcal{A}_2$; otherwise, it returns $\perp$.
**Challenge.** For time period $l^*$, $\mathcal{A}_2$ outputs a challenge set $S^*_u \subseteq \{1, \cdots, n\}$. The challenger first checks that $SK^*_{c_j}$ is the secret key corresponding to the public key $PK^*_{c_j}$. If so, it chooses a set $S^*_c \subseteq \{1, \cdots, k\}$ such that $j \in S^*_c$ and a random bit $b \in_R \{0, 1\}$, and runs $(Hdr^*, K^*) \leftarrow \mathbf{Encrypt}(PK^*, S^*_u, S^*_c, \{PK^*_i : i \in S^*\}, \{PK^*_{c_j} : j \in S^*_c\}, l^*)$. It then sets $K^*_b = K^*$, picks at random $K^*_{1-b}$ in the key space, and gives $(Hdr^*, K^*_b, K^*_{1-b})$ to the adversary $\mathcal{A}_2$. Otherwise, it gives $\perp$.
**Guess.** The adversary $\mathcal{A}_2$ outputs its guess $b' \in_R \{0, 1\}$ for $b$ and wins the game if $b = b'$ and $(l^*, PK^*_{c_j}, Hdr^*)$ was not the subject of a valid Decryption query after the Challenge.

We define $\mathcal{A}_2$'s advantage in attacking the File Sharing system in Electronic Health Records for Game 2 with parameters $(\lambda, n, k)$ as $Adv\mathsf{FSEHR}^{Game2}_{\mathcal{A}_2,n,k}(\lambda) = |Pr[b = b'] - \frac{1}{2}|$.

**Definition 1.** *We say that a File Sharing system in Electronic Health Records is adaptively $(t, \varepsilon, n, k, q_C, q_D)$-secure if no $t$-time algorithm $\mathcal{A}$ that makes at most $q_C$ Certification queries (in Game 1) and $q_D = q_{D1} + q_{D2}$ Decryption queries (in the Game 1 and Game 2 respectively), has non-negligible advantage in either Game 1 or Game 2, i.e. $Adv\mathsf{FSEHR}^{Game1}_{\mathcal{A},n,k}(\lambda) + Adv\mathsf{FSEHR}^{Game2}_{\mathcal{A},n,k}(\lambda) \leq \varepsilon$.*

We also mention the selective CCA security: a selective adversary $\mathcal{A}$ provides the set $S^*_u \subseteq \{1, \cdots, n\}$ that it wishes to be challenged on at the beginning of the security game.

We then define the Fully Collusion Resistance security, in order to capture the notion of encryption against arbitrary collusion of users. Let the number of users $n$, the number of certifiers $k$ and the security parameter $\lambda$ be given to the adversary $\mathcal{A}$ and the challenger. The game between the two entities proceeds as follows:

**Game Fully Collusion Resistant**

1. The adversary $\mathcal{A}$ outputs a set $S_{u,n'} = \{u_1, \cdots, u_{n'}\} \subseteq \{1, \cdots, n\}$ of colluding users.

2. The challenger runs the algorithm $\mathbf{Setup}(\lambda, n, k)$ to obtain the public parameters $PK$, the keys pairs $(PK_i, SK_i)$ for users $i \in S_{u,n'}$, and the keys pair $(PK_{c_j}, SK_{c_j})$ for the certifier $j \in \{1, \cdots, k\}$. It gives $(PK, \{PK_i, SK_i : i \in S_{u,n'}\}, \{PK_{c_j} : j \in \{1, \cdots, k\}\})$ to the adversary $\mathcal{A}$ and keeps $SK_{c_j}$ to itself. It also runs the algorithm $\mathbf{Certif}(PK, (PK_{c_j}, SK_{c_j}), PK_i, l)$ to obtain the certificates $Certif_{i,j,l}$ for user $i \in S_{u,n'}$, $j \in \{1, \cdots, k\}$, and time period $l$, and gives them to the adversary $\mathcal{A}$.
3. The challenger runs the algorithm $\mathbf{Encrypt}(PK, S_u, S_c, \{PK_i : i \in S_u\}, \{PK_j : j \in S_c\}, l')$ for a subset $S_u \subseteq \{1, \cdots, n\}$ such that $S_u \cap S_{u,n'} = \emptyset$ and $l' \neq l$, and for a subset $S_c \subseteq \{1, \cdots, k\}$, and gives the resulting header $Hdr$ to the adversary $\mathcal{A}$ and keeps the associated session key $K$ to itself.
4. The adversary $\mathcal{A}$ outputs $K^* \leftarrow \mathbf{Decrypt}(PK, S_u, S_c, l', (f(\{PK_i : i \in S' \subseteq S_{u,n'}\}), f(\{SK_i : i \in S' \subseteq S_{u,n'}\})), \{f(\{Certif_{i,l} : i \in S' \subseteq S_{u,n'}\}) : j \in S_c\}, Hdr^*)$ where $f$ is a function that takes as input public keys, secret keys or certificates, and outputs a new public key, a new secret key or a new certificate as a combination of public keys, secret keys or certificates, respectively.
5. The adversary $\mathcal{A}$ wins the game if $K^* = K$.

**Definition 2.** *We say that a File Sharing system in Electronic Health Records is fully collusion resistant if no $t$-time algorithm $\mathcal{A}$ has non-negligible advantage in the above game.*

### 2.3 Broadcast Encryption

A Broadcast Encryption ($\mathsf{BE}$) system [6, 5, 4, 8] is made up of three randomized algorithms $(\mathbf{Setup}_{\mathsf{BE}}, \mathbf{Encrypt}_{\mathsf{BE}}, \mathbf{Decrypt}_{\mathsf{BE}})$ such that:

1. $\mathbf{Setup}_{\mathsf{BE}}(n)$ takes as input the number of receivers $n$. It outputs $n$ secret keys $d_1, \cdots, d_n$ and a public key $PK$.
2. $\mathbf{Encrypt}_{\mathsf{BE}}(S, PK)$ takes as inputs a subset $S \subseteq \{1, \cdots, n\}$ and a public key $PK$. It outputs a pair $(Hdr, K)$ where $Hdr$ is called the header and $K \in \mathcal{K}$ is a message encryption key chosen from a finite key set $\mathcal{K}$. We will often refer to $Hdr$ as the broadcast ciphertext.
   Let $M$ be a message to be broadcast that should be decipherable precisely by the receivers in $S$. Let $C_M$ be the encryption of $M$ under the symmetric key $K$. The broadcast consists of $(S, Hdr, C_M)$. The pair $(S, Hdr)$ is often called the *full header* and $C_M$ is often called the *broadcast body*.
3. $\mathbf{Decrypt}_{\mathsf{BE}}(S, i, d_i, Hdr, PK)$ takes as inputs a subset $S \subseteq \{1, \cdots, n\}$, a user identity $i \in \{1, \cdots, n\}$ and the secret key $d_i$ for user $i$, a header $Hdr$, and the public key $PK$. If $i \in S$, then the algorithm outputs a message encryption key $K \in \mathcal{K}$. Intuitively, user $i$ can then use $K$ to decrypt the broadcast body $C_M$ and obtain the message body $M$.

We require that for all subset $S \subseteq \{1, \cdots, n\}$ and all $i \in S$, if $(PK, (d_1, \cdots, d_n)) \leftarrow \mathbf{Setup}_{\mathsf{BE}}(n)$ and $(Hdr, K) \leftarrow \mathbf{Encrypt}_{\mathsf{BE}}(S, PK)$, then $\mathbf{Decrypt}_{\mathsf{BE}}(S, i, d_i, Hdr, PK) = K$.

### 2.4 Certificate-Based Encryption

A certificate-updating Certificate-Based Encryption ($\mathsf{CBE}$) system is made up of six randomized algorithms ($\mathbf{Gen}_{\mathsf{IBE}}, \mathbf{Gen}_{\mathsf{PKE}}, \mathbf{Upd1}, \mathbf{Upd2}, \mathbf{Encrypt}_{\mathsf{CBE}}, \mathbf{Decrypt}_{\mathsf{CBE}}$) such that:

1. $\mathbf{Gen}_{\mathsf{IBE}}(\lambda_1, t)$ takes as inputs a security parameter $\lambda_1$ and (optionally) the total number of time periods $t$. It outputs $SK_{\mathsf{IBE}}$ (the certifier's master secret key) and public parameters $params$ that include a public key $PK_{\mathsf{IBE}}$ and the description of a string space $\mathcal{S}$.
2. $\mathbf{Gen}_{\mathsf{PKE}}(\lambda_2, t)$ takes as inputs a security parameter $\lambda_2$ and (optionally) the total number of time periods $t$. It outputs $SK_{\mathsf{PKE}}$ and public key $PK_{\mathsf{PKE}}$ (the client's secret and public keys).
3. $\mathbf{Upd1}(SK_{\mathsf{IBE}}, params, l, s, PK_{\mathsf{PKE}})$ takes as inputs $SK_{\mathsf{IBE}}$, $params$, $l$, string $s \in \mathcal{S}$ and $PK_{\mathsf{PKE}}$, at the start of time period $l$. It outputs $Cert'_l$, which is sent to the client.
4. $\mathbf{Upd2}(params, l, Cert'_l, Cert_{l-1})$ takes as inputs $params$, $l$, $Cert'_l$ and (optionally) $Cert_{l-1}$, at the start of time period $l$. It outputs $Cert_l$.
5. $\mathbf{Encrypt}_{\mathsf{CBE}}(params, l, s, PK_{\mathsf{PKE}}, M)$ takes as inputs ($params, l, s, PK_{\mathsf{PKE}}$, $M$), where $M$ is a message. It outputs a ciphertext $C$ on message $M$ intended for the client to decrypt using $Cert_l$ and $SK_{\mathsf{PKE}}$ (and possibly $s$).
6. $\mathbf{Decrypt}_{\mathsf{CBE}}(params, Cert_l, SK_{\mathsf{PKE}}, C, l)$ takes as inputs ($params, Cert_l, SK_{\mathsf{PKE}}$, $C$) in time period $l$. It outputs either $M$ or the special case $\perp$ indicating failure.

We require that $\mathbf{Decrypt}(params, Cert_l, SK_{\mathsf{PKE}}, \mathbf{Encrypt}(params, l, s, PK_{\mathsf{PKE}}, M), l) = M$ for the given $params \leftarrow \mathbf{Gen}_{\mathsf{IBE}}(\lambda_1, t)$, $PK \leftarrow \mathbf{Gen}_{\mathsf{PKE}}(\lambda_2, t)$, $Cert_l \leftarrow \mathbf{Upd2}(params, l, \mathbf{Upd1}(SK_{\mathsf{IBE}}, params, l, s, PK_{\mathsf{PKE}}), Cert_{l-1})$.

## 3 An Efficient Construction

Our construction $\mathsf{FSEHR}$ is an effective combination of the BGW Broadcast Encryption ($\mathsf{BE}$) scheme [4] and the Gentry's Certificate-Based Encryption ($\mathsf{CBE}$) scheme [7]. Notice that the Gentry's $\mathsf{CBE}$ scheme is designed for a communication between one sender and one receiver. Therefore, applying the Gentry's $\mathsf{CBE}$ scheme directly to the BGW, will lead to the linear size of the headers in the number of users and of certifiers in the two respective subsets designed by the sender, which is impractical. However, we managed to overcome this issue and achieve constant size for header, secret keys and certificates. Moreover, our scheme $\mathsf{FSEHR}$ is proved selective CCA secure using the standard transformation from the REACT scheme proposed by Okamoto and Pointcheval [11].

**Setup**$(\lambda, n, k)$. On input the security parameter $\lambda$, the total number $n$ of users, and the total number $k$ of certifiers, run $(p, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathbf{GroupGen}(\lambda, n, k)$. Pick at random $g \in_R \mathbb{G}$ and $\alpha \in_R \mathbb{Z}_p$, compute $g_i = g^{(\alpha^i)}$ for $i = 1, \cdots, n, n+2, \cdots, 2n$. Pick at random $\gamma \in_R \mathbb{Z}_p$ and compute $v = g^\gamma$. Choose three hash functions

$H_1 : \mathbb{G} \times \{0,1\}^* \to \mathbb{G}$, $H_2 : \mathbb{G} \times \mathbb{G} \to \mathbb{G}$, and $H_3 : \mathbb{G}_T \times \mathbb{G} \times \mathbb{G} \times \mathbb{G} \times \mathbb{G} \to \{0,1\}^\lambda$. For user $i \in \{1, \cdots, n\}$, compute the user's secret key as $d_i = g_i^\gamma (= v^{(\alpha^i)})$.

Independently, for $j \in \{1, \cdots, k\}$, certifier $j$ computes its own public/secret key pair as follows: choose at random an exponent $\sigma_j \in_R \mathbb{Z}_p$ and then compute the public key $w_j = g^{\sigma_j}$. Set the secret key as $d_{c_j} = \sigma_j$.

Set the public parameters as $PK = (p, \mathbb{G}, \mathbb{G}_T, g, g_1, \cdots, g_n, g_{n+2}, \cdots, g_{2n}, v, w_1, \cdots, w_k, H_1, H_2, H_3)$.

**Certif**$(PK, d_{c_j}, i, l)$. On input the public parameters $PK$, the certifier $j$'s secret key $d_{c_j}$, the user $i$, and the time period $l$ represented as a string in $\{0,1\}^*$, pick at random $r_{i,j,l} \in_R \mathbb{Z}_p$ and compute the user's certificate $e_{i,j,l} = (e_{i,j,l,1}, e_{i,j,l,2})$ as follows:

$$e_{i,j,l,1} = g_i^{\sigma_j} \cdot H_1(w_j, l)^{\sigma_j \cdot r_{i,j,l}} (= w_j^{(\alpha^i)} \cdot H_1(w_j, l)^{\sigma_j \cdot r_{i,j,l}})$$
$$e_{i,j,l,2} = g^{\sigma_j \cdot r_{i,j,l}} (= w_j^{r_{i,j,l}})$$

**Encrypt**$(PK, S_u, S_c, l)$. On input the public parameters $PK$, a set $S_u \subseteq \{1, \cdots, n\}$ of users, a set $S_c \subseteq \{1, \cdots, k\}$ of certifiers, and the time period $l$, pick at random an exponent $t \in_R \mathbb{Z}_p$, compute the session key $K = e(g_{n+1}, g)^t$, and set the header $Hdr = (C_1, C_2, C_3, C_4, C_5)$ as follows:

$$(g^t, \prod_{j \in S_c} H_1(w_j, l)^t, (v \cdot \prod_{j \in S_c} w_j \cdot \prod_{i' \in S_u} g_{n+1-i'})^t, H_2(C_1, C_3)^t, H_3(K, C_1, C_2, C_3, C_4))$$

**Decrypt**$(PK, S_u, S_c, l, i, d_i, e_{i,j,l}, Hdr)$. On input the public parameters $PK$, a set $S_u \subseteq \{1, \cdots, n\}$ of users, a set $S_c \subseteq \{1, \cdots, k\}$ of certifiers, the time period $l$, the user $i$ with its secret key $d_i$ and its certificates $e_{i,j,l}$ for $j \in S_c$, parsed as $(e_{i,j,l,1}, e_{i,j,l,2})$, and the header $Hdr$ parsed as $(C_1, C_2, C_3, C_4, C_5)$, check whether $e(C_1, H_2(C_1, C_3)) \stackrel{?}{=} e(g, C_4)$, and output

$$K = \frac{e(g_i, C_3) \cdot e(\prod_{j \in S_c} e_{i,j,l,2}, C_2)}{e(d_i \cdot \prod_{j \in S_c} e_{i,j,l,1} \cdot \prod_{i' \in S_u \setminus \{i\}} g_{n+1-i'+i}, C_1)} = e(g_{n+1}, g)^t$$

Then, compute $C_5' = H_3(K, C_1, C_2, C_3, C_4)$. If $C_5' = C_5$, then return $K$; otherwise return $\perp$.

**Correctness.** Notice that $g_i^{(\alpha^{i'})} = g_{i+i'}$ for any $i, i'$. At time period $l$, user $i \in S_u \subseteq \{1, \cdots, n\}$ with secret key $d_i$ and certificate $e_{i,l,j}$ for $j \in S_c \subseteq \{1, \cdots, k\}$ decrypts as follows:

$$\begin{aligned} K &= \frac{e(g_i, C_3) \cdot e(\prod_{j \in S_c} e_{i,j,l,2}, C_2)}{e(d_i \cdot \prod_{j \in S_c} e_{i,j,l,1} \cdot \prod_{i' \in S_u \setminus \{i\}} g_{n+1-i'+i}, C_1)} \\ &= \frac{e(g^{(\alpha^i)}, (v \cdot \prod_{j \in S_c} w_j \cdot \prod_{i' \in S_u} g_{n+1-i'})^t) \cdot e(\prod_{j \in S_c} g^{\sigma_j \cdot r_{i,j,l}}, \prod_{j \in S_c} H_1(w_j, l)^t)}{e(v^{(\alpha^i)} \cdot \prod_{j \in S_c} w_j^{(\alpha^i)} \cdot H_1(w_j, l)^{\sigma_j \cdot r_{i,j,l}} \cdot \prod_{i' \in S_u \setminus \{i\}} g_{n+1-i'+i}, g^t)} \\ &= e(g_{n+1}, g)^t \end{aligned}$$

**Performance.** In the following table, we evaluate the efficiency of our scheme FSEHR. We use results of cryptographic operation implementations (exponentiations and pairings) using the MIRACL library, provided by Certivox for the MIRACL Authentication Server Project Wiki. All the following experiments are based on Borland C/C++ Compiler/Assembler and tested on a processor 2.4 GHz Intel i5 520M.

For our symmetric pairing-based systems, AES with a 80-bit key and a Super Singular curve over $\mathbb{GF}_p$, for a 512-bit modulus $p$ and an embedding degree equal to 2, are used. We assume that there are $n = 100$ users and $k = 20$ certifiers.

| | | Exponentiation in $\mathbb{G}$ | Exponentiation in $\mathbb{G}_T$ | Pairings |
|---|---|---|---|---|
| Time/computation | | 1.49 | 0.36 | 3.34 |
| Algorithms | **Setup** | 546,80 | | |
| | **Certif** | 89,40 | | |
| | **Encrypt** | 5,96 | 0,36 | 3,34 |
| | **Decrypt** | | | 16,70 |

**Fig. 2.** Timings for our symmetric pairing-based system FSEHR. Times are in milliseconds.

We note that the total time in the algorithm **Setup** is substantial, but we recall that this algorithm should be run only once to generate the public parameters and the static secret keys for both users and certifiers. In the algorithm **Certif**, it requires 89,40 milliseconds because $k = 20$ certificates are created. Finally, in the algorithms **Encrypt** and **Decrypt**, it takes 9,66 and 16,70 milliseconds respectively, mainly due to the cost of pairing computations.

## 4 Security Proofs

**Assumption.** We prove the security of our scheme FSEHR using the Decisional $n$-Bilinear Diffie-Hellman Exponent (DBDHE) assumption, which is as follows.

**Definition 3.** *The $(t, n, \varepsilon)$-BDHE assumption says that for any $t$-time adversary $\mathcal{B}$ that is given $(g, h, g^a, g^{a^2}, \cdots, g^{a^n}, g^{a^{n+2}}, \cdots, g^{a^{2n}}) \in \mathbb{G}^{2n+1}$, and a candidate to Decisional $n$-BDHE problem, that is either $e(g,h)^{a^{n+1}} \in \mathbb{G}_T$ or a random value $T$, cannot distinguish the two cases with advantage greater than $\varepsilon$:*

$$AdvBDHE_{\mathcal{B},n} = |Pr[\mathcal{B}(g, h, g^a, g^{a^2}, \cdots, g^{a^n}, g^{a^{n+2}}, \cdots, g^{a^{2n}}, e(g,h)^{a^{n+1}}) = 1] - Pr[\mathcal{B}(g, h, g^a, g^{a^2}, \cdots, g^{a^n}, g^{a^{n+2}}, \cdots, g^{a^{2n}}, T) = 1]| \leq \varepsilon$$

**Selective CCA Security Proof.**

**Theorem 1.** *The File Sharing scheme in Electronic Health Records* FSEHR *achieves Selective CCA Security under the Decision $n$-BDHE assumption, in the random oracle model.*

*Proof.* We assume there exists an adversary $\mathcal{A}$ that breaks the semantic security of the FSEHR scheme with probability greater than $\varepsilon$ within time $t$, making $q_{H_1}$, $q_{H_2}$ and $q_{H_3}$ random oracle queries, $q_{cf}$ certification queries and $q_d$ decryption queries. Using this adversary $\mathcal{A}$, we build an attacker $\mathcal{B}$ for the Decisional $n$-BDHE problem in $\mathbb{G}$. $\mathcal{B}$ proceeds as follows.

For simplicity, we write $g_i = g^{(\alpha^i)}$ for an implicitly defined $\alpha$. $\mathcal{B}$ first takes as input a Decisional $n$-BDHE intance $(g, h, g_1, \cdots, g_n, g_{n+2}, \cdots, g_{2n}, Z)$ where $Z = e(g_{n+1}, h)$ or $Z \in_R \mathbb{G}_T$. $\mathcal{B}$ makes use of three random oracles $H_1$, $H_2$ and $H_3$, and three respective hash lists $L_1$, $L_2$ and $L_3$, initially set empty, to store all the query-answers.

**Init.** $\mathcal{A}$ outputs a set $S_u^* \subseteq \{1, \cdots, n\}$ of users that it wishes to be challenged on.

**Setup.** $\mathcal{B}$ needs to generate the public parameters $PK$, the secret keys $d_i$ for $i \notin S_u^*$, and the secret keys $d_{c_j}$ for the certifier $j \in \{1, \cdots, k\}$. It chooses random elements $x, y_1 \cdots, y_k \in_R \mathbb{Z}_p$, and sets $v = g^x / \prod_{i' \in S_u^*} g_{n+1-i'}$, $w_j = g^{y_j}$ and $d_{c_j} = y_j$ for $j \in \{1, \cdots, k\}$. It then computes

$$d_i = g_i^x / \prod_{i' \in S_u^*} g_{n+1-i'+i} = g^{x \cdot (\alpha^i)} \cdot (\prod_{i' \in S_u^*} g_{n+1-i'})^{-(\alpha^i)} = v^{(\alpha^i)}.$$

Eventually, $\mathcal{B}$ gives $\mathcal{A}$ the public parameters $PK = (p, \mathbb{G}, \mathbb{G}_T, e, g, g_1, \cdots, g_n, g_{n+2}, \cdots, g_{2n}, v, w_1, \cdots, w_k, H_1, H_2, H_3)$, where $H_1$, $H_2$ and $H_3$ are controlled by $\mathcal{B}$ as follows:

- Upon receiving a query $(w_j, l_j)$ to the random oracle $H_1$ for some $j \in [1, q_{H_1}]$:
  - If $((w_j, l_j), u_j, U_j)$ exists in $L_1$, return $U_j$.
  - Otherwise, choose $u_j \in_R \mathbb{Z}_p$ at random and compute $U_j = g^{u_j}$. Put $((w_j, l_j), u_j, U_j)$ in $L_1$ and return $U_j$ as answer.
- Upon receiving a query $(W_{1j}, W_{2j})$ to the random oracle $H_2$ for some $j \in [1, q_{H_2}]$:
  - If $((W_{1j}, W_{2j}), X_j)$ exists in $L_2$, return $X_j$.
  - Otherwise, choose $X_j \in_R \mathbb{G}$ at random. Put $((W_{1j}, W_{2j}), X_j)$ in $L_2$ and return $X_j$ as answer.
- Upon receiving a query $(K_j, C_{1j}, C_{2j}, C_{3j}, C_{4j})$ to the random oracle $H_3$ for some $j \in [1, q_{H_3}]$:
  - If $((K_j, C_{1j}, C_{2j}, C_{3j}, C_{4j}), C_{5j})$ exists in $L_3$, return $C_{5j}$.
  - Otherwise, choose $C_{5j} \in_R \{0,1\}^\lambda$ at random. Put $(K_j, C_{1j}, C_{2j}, C_{3j}, C_{4j})$, $C_{5j})$ in $L_3$ and return $C_{5j}$ as answer.

**Phase 1.** $\mathcal{B}$ answers $\mathcal{A}$'s queries as follows.

1. Upon receiving a Certification query $(PK_{i'}, d_i, l_{i'})$ for some $i \in \{1, \cdots, n\}$ and $i' \in [1, q_{cf}]$:
   - If $((w_{i',j}, l_{i'}), u_{i',j}, U_{i',j})$ exists in $L_1$, return the pair $(u_{i',j}, U_{i',j})$ for $j \in \{1, \cdots, k\}$.
   - Otherwise, choose $u_{i',j} \in_R \mathbb{Z}_p$ at random, compute $U_{i',j} = g^{u_{i',j}}$, and put $((w_{i',j}, l_{i'}), u_{i',j}, U_{i',j})$ in $L_1$.

Then, pick at random $r_{i,j,l_{i'}} \in_R \mathbb{Z}_p$ and return the certificate $e_{i,j,l_{i'}}$ where

$$\begin{aligned} e_{i,j,l_{i'},1} &= g_i^{y_j} \cdot (U_{i',j}^{y_j})^{r_{i,j,l_{i'}}} = g^{y_j \cdot (\alpha^i)} \cdot (g^{u_{i',j}})^{r_{i,j,l_{i'}} \cdot y_j} \\ &= g^{y_j \cdot (\alpha^i)} \cdot H_1(w_{i',j}, l_{i'})^{r_{i,j,l_{i'}} \cdot y_j} = w_{i',j}^{(\alpha^i)} \cdot w_{i',j}^{r_{i,j,l_{i'}} \cdot u_{i',j}} \\ e_{i,j,l_{i'},2} &= w_{i',j}^{r_{i,j,l_{i'}}} \end{aligned}$$

2. Upon receiving a Decryption query $(S_{u,i'}, Hdr_{i'}, i, l_{i'})$ for some $i \in \{1, \cdots, n\}$ and $i' \in [1, q_d]$, where $Hdr_{i'} = (C_{1i'}, C_{2i'}, C_{3i'}, C_{4i'}, C_{5i'})$:
   - If $((K_{i'}, C_{1i'}, C_{2i'}, C_{3i'}, C_{4i'}), C_{5i'})$ exists in $L_3$, do the following:
     - Compute $H_1(w_{i',j}, l_{i'})$ using the simulation of $H_1$ as above and check whether $e(C_{2i'}, g) \stackrel{?}{=} e(C_{1i'}, \prod_{j \in S_c} H_1(w_{i',j}, l_{i'}))$ for $j \in S_c \subseteq \{1, \cdots, k\}$. If not, return $\perp$. Otherwise, compute $H_2(C_{1i'}, C_{3i'})$ using the simulation of $H_2$ as above and check whether $e(C_{1i'}, H_2(C_{1i'}, C_{3i'})) \stackrel{?}{=} e(g, C_{4i'})$. If not, return $\perp$. Otherwise, check whether $K_{i'}$ is equal or not to
     $$\frac{e(g_i, C_{3i'}) \cdot e(g, C_{2i'})}{e(g_i^{x + sum_{j \in S_c} y} \cdot \prod_{j \in S_c} H_1(w_{i',j}, l_{i'}), C_{1i'})}.$$
     If the above equation holds, return $K_{i'}$. Otherwise, return $\perp$.
   - Else, return $\perp$.

**Challenge.** $\mathcal{B}$ generates the challenge on the challenge set $S_u^*$ as follows. First, $\mathcal{B}$ sets $C_1^* = h$ and searchs in $L_1$ to get $u$ that corresponds to $(w_j, l)$ such that $j \in S_c \subseteq \{1, \cdots, k\}$. Then, it computes $C_2^* = h^u$. It also computes $C_3^* = h^{x + \sum_{j \in S_c} y}$. Informally, we write $h = g^t$ for an unknown $t \in \mathbb{Z}_p$. Then, it randomly chooses an exponent $z \in_R \mathbb{Z}_p$, and sets $C_4^* = h^z = H_2(C_1^*, C_3^*)^t$ . Second, it randomly chooses a bit $b \in_R \{0,1\}$ and sets $K_b = Z$ and picks a random $K_{b-1}$ in $\mathbb{G}_T$. It also picks $C_5^* \in_R \{0,1\}^*$ at random, and sets $C_5^* = H_3(K_b, C_1^*, C_2^*, C_3^*, C_4^*)$. Finally, it returns $(Hdr^* = (C_1^*, C_2^*, C_3^*, C_4^*, C_5^*), K_0, K_1)$ as the challenge to $\mathcal{A}$.

When $Z = e(g_{n+1}, h)$, then $(Hdr^*, K_0, K_1)$ is a valid challenge for $\mathcal{A}$'s point of view, as in the real attack. Indeed, if we write $h = g^t$ for an unknown $t \in \mathbb{Z}_p$, then

$$\begin{aligned} h^{x + \sum_{j \in S_c} y} &= h^x \cdot \prod_{j \in S_c} h^{y_j} \cdot \left( \frac{\prod_{i' \in S_u^*} g_{n+1-i'}}{\prod_{j \in S_u^*} g_{n+1-i'}} \right)^t = g^{t \cdot x} \cdot \prod_{j \in S_c} g^{t \cdot y_j} \cdot \left( \frac{\prod_{i' \in S_u^*} g_{n+1-i'}}{\prod_{j \in S_u^*} g_{n+1-i'}} \right)^t \\ &= \left( \frac{g^x}{\prod_{i' \in S_u^*} g_{n+1-i'}} \cdot \prod_{j \in S_c} g^{y_j} \cdot \prod_{i' \in S_u^*} g_{n+1-i'} \right)^t = C_3^* \end{aligned}$$

Therefore, by definition, $Hdr^*$ is a valid encryption of the session key $e(g_{n+1}, g)^t$. Moreover, $e(g_{n+1}, g)^t = e(g_{n+1}, h) = Z = K_b$, and thus $(Hdr^*, K_0, K_1)$ is a valid challenge to $\mathcal{A}$. When $Z \in_R \mathbb{G}_T$, then $K_0$ and $K_1$ are random independent elements in $\mathbb{G}_T$.

**Phase 2.** $\mathcal{B}$ responds to the Certification and Decryption queries as in **Phase 1**. We note that if $(K_b^*, C_1^*, C_2^*, C_3^*, C_4^*)$ is asked to the random oracle $H_3$, the value $C_5^*$ created in the simulation of the Challenge is returned.

**Guess.** The adversary $\mathcal{A}$ outputs its guess $b' \in_R \{0,1\}$ for $b$.

**Analysis.** In the simulations of the secret key generation and the certificate generation, the responses to $\mathcal{A}$ are perfect.

The simulation of the random oracle $H_1$ is not entirely perfect. Let $\mathbf{Query}H_1$ be the event that $\mathcal{A}$ has queried before the **Challenge** phase $(w_j^*, l^*)$ to $H_1$ for $j \in \{1, \cdots, k\}$. Except for the case above, the simulation of $H_1$ is perfect. This event happens with probability $k/p$.

The simulation of the random oracles $H_2$ and $H_3$ are not entirely perfect. Let $\mathbf{Query}H_2$ and $\mathbf{Query}H_3$ be the events that $\mathcal{A}$ has queried before the **Challenge** phase $(C_1^*, C_3^*)$ to $H_2$ and $(K_b^*, C_1^*, C_2^*, C_3^*, C_4^*)$ to $H_3$. Except for the cases above, the simulations of $H_2$ and $H_3$ are perfect. These two events happen with probability $1/p$ and $1/2^\lambda$ respectively.

The simulation of the Decryption oracle is nearly perfect, but a valid header can be rejected sometimes. Indeed, in the simulation of the Decryption oracle, if $(K, C_1, C_2, C_3, C_4)$ has not been queried to $H_3$, then the header is rejected. This leads to two cases:

1. $\mathcal{A}$ uses the value $C_5^*$, which is part of the challenge, as a part of its Decryption query.
2. $\mathcal{A}$ has guessed a right value for the output of $H_3$ without querying it.

However, in the first case, since $(C_1^*, C_2^*, C_3^*, C_4^*)$ and $K_b$ are provided as input to $H_3$, the Decryption query that $\mathcal{A}$ would ask is the same as the challenge, which is not allowed to query. The second case may happen but with negligible probability $1/2^k$. Let $\mathbf{DecO}$ denote the event that $\mathcal{A}$ correctly guesses the output of $H_3$. Therefore, if $\mathcal{B}$ does not correctly guess the output of $H_3$, $\mathcal{A}$'s point of view in the simulation is identical to the one in the real attack.

Thus, we have $Pr[\mathcal{B}(g, g_1, \cdots, g_n, g_{n+2}, \cdots, g_{2n}, h) = e(g_{n+1}, h)] = |Pr[b' = b | \neg\mathbf{Query}H_1 \wedge \neg\mathbf{Query}H_2 \wedge \neg\mathbf{Query}H_3 \wedge \neg\mathbf{DecO}] - \frac{1}{2}|$ By definition of $\mathcal{A}$, we have $|Pr[b' = b] - \frac{1}{2}| > \varepsilon - Pr[\mathbf{Query}H_1] - Pr[\mathbf{Query}H_2] - Pr[\mathbf{Query}H_3] - Pr[\mathbf{DecO}]$. Hence, we have

$$
\begin{aligned}
&|Pr[b' = b | \neg\mathbf{Query}H_1 \wedge \neg\mathbf{Query}H_2 \wedge \neg\mathbf{Query}H_3 \wedge \neg\mathbf{DecO}] - \frac{1}{2}| \\
&> |Pr[b' = b] - Pr[\mathbf{Query}H_1] - Pr[\mathbf{Query}H_2] - Pr[\mathbf{Query}H_3] - Pr[\mathbf{DecO}] - \frac{1}{2}| \\
&> \varepsilon - Pr[\mathbf{Query}H_1] - Pr[\mathbf{Query}H_2] - Pr[\mathbf{Query}H_3] - Pr[\mathbf{DecO}]
\end{aligned}
$$

Since $\mathcal{A}$ makes $q_{H_1}$, $q_{H_2}$ and $q_{H_3}$ random oracle queries during the attack, $Pr[\mathbf{Query}H_1] \leq kq_{H_1}/p$, $Pr[\mathbf{Query}H_2] \leq q_{H_2}/p$ and $Pr[\mathbf{Query}H_3] \leq q_{H_3}/2^\lambda$. In the same way, since $\mathcal{A}$ makes $q_d$ Decryption queries during the attack, $Pr[\mathbf{DecO}] \leq q_d/2^\lambda$. Therefore, we have $\mathcal{B}$'s winning probability $\varepsilon' > \varepsilon - \frac{k \cdot q_{H_1} + q_{H_2}}{p} - \frac{q_d + q_{H_3}}{2^\lambda}$.

**Fully Collusion Resistance Proof.**

**Theorem 2.** *The File Sharing scheme in Electronic Health Records* FSEHR *is fully secure against any number of colluders, in the random oracle model.*

*Proof.* We assume there exists an adversary $\mathcal{A}$ that breaks the semantic security of the FSEHR scheme with probability greater than $\varepsilon$ when interacting with an algorithm $\mathcal{B}$.

$\mathcal{A}$ chooses a subset $S_{u,n'} = \{u_1, \cdots, u_{n'}\} \subseteq \{1, \cdots, n\}$ of colluding users. $\mathcal{B}$ then runs $\mathbf{Setup}(\lambda, n, k)$, provides the public parameters $PK = (p, \mathbb{G}, \mathbb{G}_T, g, g_1, \cdots, g_n, g_{n+2}, \cdots, g_{2n}, v, w_1, \cdots, w_k, H_1, H_2, H_3)$, the secret keys $d_i = v^{(\alpha^i)}$ for user $i \in S_{u,n'}$ to $\mathcal{A}$, and keeps secret the certifier's secret keys $d_{c_j} = \sigma_j$ for $j \in \{1, \cdots, k\}$. It also runs $\mathbf{Certif}(PK, d_{c_j}, i, l)$ and gives the certificate $e_{i,j,l} = (e_{i,j,l,1} = w_j^{(\alpha^i)} \cdot H_1(w_j, l)^{\sigma_j \cdot r_{i,j,l}}, e_{i,j,l,2} = w_j^{r_{i,j,l}})$ to $\mathcal{A}$ for user $i \in S_{u,n'}$, $j \in \{1, \cdots, k\}$, and time period $l$.

Afterwards, $\mathcal{B}$ chooses a subset $S \subseteq \{1, \cdots, n\}$ of users such that $S \cap S_{u,n'} = \emptyset$, time period $l' \neq l$, and set $S_c \subseteq \{1, \cdots, k\}$. It outputs $(Hdr, K) \leftarrow \mathbf{Encrypt}(PK, S, S_c, l)$, gives $Hdr = (C_1 = g^t, C_2 = \prod_{j \in S_c} H_1(w_j, l)^t, C_3 = (v \cdot \prod_{j \in S_c} w_j \cdot \prod_{i' \in S} g_{n+1-i'})^t, C_4 = H_2(C_1, C_3)^t, C_5 = H_3(K, C_1, C_2, C_3, C_4))$ to $\mathcal{A}$ and keeps secret the session key $K = e(g_{n+1}, g)^t$.

$\mathcal{A}$ computes new secret key $d_\chi$ and certificate $e_{\chi,j,l}$ from the secret keys and certificates of users in $S_{u,n'}$ that it previously obtained as follows. First, $\mathcal{A}$ defines a subset $S_{u,\bar{n}} \subseteq S_{u,n'}$, and sets

$$
\begin{aligned}
d_\chi &= \prod_{i' \in S_{u,\bar{n}}} d_{i'} = v^{\sum_{i' \in S_{u,\bar{n}}} \alpha^{i'}} = v^{f_\chi(\alpha)} \\
e_{\chi,j,l,1} &= \prod_{i' \in S_{u,\bar{n}}} e_{i',j,l,1} = w_j^{\sum_{i' \in S_{u,\bar{n}}} \alpha^{i'}} \cdot H_1(w_j, l)^{\sigma_j \cdot \sum_{i' \in S_{u,\bar{n}}} r_{i',j,l}} \\
&= w_j^{f_\chi(\alpha)} \cdot H_1(w_j, l)^{\sigma_j \cdot r_{\chi,j,l}} \\
e_{\chi,j,l,2} &= w_j^{\sum_{i' \in S_{u,\bar{n}}} r_{i',j,l}} = w_j^{r_{\chi,j,l}}
\end{aligned}
$$

where $f_\chi(\alpha) = \sum_{i' \in S_{u,\bar{n}}} \alpha^{i'}$. For a user $i$ belonging to $S$, it has to cancel out the following terms

$$
\frac{e(g, (v^{(\alpha^i)} \cdot \prod_{j \in S_c} w_j^{(\alpha^i)} \cdot \prod_{i' \in S \setminus \{i\}} g_{n+1-i'+i})^t)}{e(v^{f_\chi(\alpha)} \cdot \prod_{j \in S_c} w_j^{f_\chi(\alpha)} \cdot \prod_{i' \in S \setminus \{i\}} g_{n+1-i'+i}, g^t)} = \frac{e(g, v^{(\alpha^i)} \cdot \prod_{j \in S_c} w_j^{(\alpha^i)})}{e(v^{f_\chi(\alpha)} \cdot \prod_{j \in S_c} w_j^{f_\chi(\alpha)}, g)}
$$

$$
\frac{\prod_{j \in S_c} e(g^{\sigma_j \cdot r_{\chi,j,l}}, \prod_{j \in S_c} H_1(w_j, l')^t)}{\prod_{j \in S_c} e(H_1(w_j, l)^{\sigma \cdot r_{\chi,j,l}}, g^t)} = \frac{e(g, \prod_{j \in S_c} H_1(w_j, l'))}{e(\prod_{j \in S_c} H_1(w_j, l), g)}
$$

In the first equality, the two terms $v^{\alpha^i - f_\chi(\alpha)}$ and $w_j^{\alpha^i - f_\chi(\alpha)}$ should be wiped out. Thus, one wants to have $f_\chi(\alpha) = \sum_{i' \in S_{u,\bar{n}}} \alpha^{i'} = \alpha^i \bmod p$. In other words, $\alpha$ is

one root of the polynomial $P(x) = \sum_{i' \in S_{u,\bar{n}} \cup \{i\}} x^{i'}$ of degree $\bar{n}+1$. This happens with probability $Pr[f_\chi(\alpha) = \alpha^i \bmod p] \leq (\bar{n}+1)/p$. In the second equality, if we suppose that $l \neq l'$, $j, j' \in \{1, \cdots, k\}$, and the hash function $H_1$ is a random oracle, then the probability that the two outputs are equal $Pr[H_1(w_{j'}, l') = H_1(w_j, l)] \leq 1/p$. Finally, $\mathcal{A}$ has negligible advantage $Adv\mathsf{FSEHR}_{\mathcal{A},n,k} \leq (\bar{n}+2)/p$ to retrieve the session key $K$.

## References


1. J. A. Akinyele, M. W. Pagano, M. D. Green, C. U. Lehmann, Z. N. Peterson, and A. D. Rubin. Securing electronic medical records using attribute-based encryption on mobile devices. In *SPSM '11*, ACM, pp 75-86, 2011.
2. J. L. F. Alemán, I. C. Señor, P. A. O. Lozoya, and A. Toval. Security and privacy in electronic health records: A systematic literature review. *Journal of Biomedical Informatics*, 46(3):541–562, 2013.
3. J. Benaloh, M. Chase, E. Horvitz, and K. Lauter. Patient controlled encryption: Ensuring privacy of electronic medical records. In *CCSW '09*, ACM, pp 103-114, 2009.
4. D. Boneh, C. Gentry, and B. Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In *CRYPTO'05*, LNCS, pp 258-275. Springer-Verlag, 2005.
5. Y. Dodis and N. Fazio. Public key broadcast encryption for stateless receivers. In *Digital Rights Management*, LNCS 2696, pp 61-80. Springer, 2003.
6. A. Fiat and M. Naor. Broadcast encryption. In *CRYPTO '93*, LNCS, pp 480-491. Springer-Verlag, 1994.
7. C. Gentry. Certificate-based encryption and the certificate revocation problem. In *EUROCRYPT'03*, LNCS, pp 272-293. Springer-Verlag, 2003.
8. C. Gentry and B. Waters. Adaptive security in broadcast encryption systems (with short ciphertexts). In *EUROCRYPT 2009*, LNCS 5479, pp 171-188. Springer, 2009.
9. G. H. M. B. Motta and S. S. Furuie. A contextual role-based access control authorization model for electronic patient record. *IEEE Transactions on Information Technology in Biomedicine*, 7(3):202–207, 2003.
10. S. Narayan, M. Gagné, and R. Safavi-Naini. Privacy preserving ehr system using attribute-based infrastructure. In *CCSW '10*, ACM, pp 47-52, 2010.
11. T. Okamoto and D. Pointcheval. React: Rapid enhanced-security asymmetric cryptosystem transform. In *CT-RSA 2001*, LNCS 2020, pp 159-174. Springer, 2001.
12. M. Peleg, D. Beimel, D. Dori, and Y. Denekamp. Situation-based access control: Privacy management via modeling of patient data access scenarios. *J. of Biomedical Informatics*, 41(6):1028–1040, Dec. 2008.
13. H. Wang, Q. Wu, B. Qin, and J. Domingo-Ferrer. Frr: Fair remote retrieval of outsourced private medical records in electronic health networks. In *J Biomed Inform*. Elsevier, 2014.