Hindawi Publishing Corporation
Mobile Information Systems
Volume 2016, Article ID 6545873, 13 pages
http://dx.doi.org/10.1155/2016/6545873

Research Article

Efficient Attribute-Based Secure Data Sharing with Hidden Policies and Traceability in Mobile Health Networks

Changhee Hahn, Hyunsoo Kwon, and Junbeom Hur

Department of Computer Science and Engineering, Korea University, Seoul 136-701, Republic of Korea

Correspondence should be addressed to Junbeom Hur; jbhur@korea.ac.kr

Received 26 November 2015; Accepted 12 June 2016

Academic Editor: Wenyao Xu

Mobile health (also written as mHealth) provisions the practice of public health supported by mobile devices. mHealth systems let patients and healthcare providers collect and share sensitive information, such as electronic and personal health records (EHRs) at any time, allowing more rapid convergence to optimal treatment. Key to achieving this is securely sharing data by providing enhanced access control and reliability. Typically, such sharing follows policies that depend on patient and physician preferences defined by a set of attributes. In mHealth systems, not only the data but also the policies for sharing it may be sensitive since they directly contain sensitive information which can reveal the underlying data protected by the policy. Also, since the policies usually incur linearly increasing communication costs, mHealth is inapplicable to resource-constrained environments. Lastly, access privileges may be publicly known to users, so a malicious user could illegally share his access privileges without the risk of being traced. In this paper, we propose an efficient attribute-based secure data sharing scheme in mHealth. The proposed scheme guarantees a hidden policy, constant-sized ciphertexts, and traces, with security analyses. The computation cost to the user is reduced by delegating approximately 50% of the decryption operations to the more powerful storage systems.

1. Introduction

mHealth is an abbreviation for mobile health, which can encompass a wide range of healthcare technologies such as mobile computing, medical sensors, and communication technologies [1]. Rapid growth in wireless communications, availability and miniaturization of mobile devices, and computing resources in parallel with mobile and wearable systems can boost the wide adoption of mHealth. Such developments can greatly impact on and reshape the processes of existing healthcare services. For instance, semiconductor-implanted smart intelligent sensors will allow drugs to be delivered in real time to a personal server when they sense a patient who needs a dose of drugs. Personal servers, such as mobile devices, supply global connectivity to the storage center, which can thereby serve clinical healthcare from a distance [2]. The storage center holds the information that forms the electronic health record (EHR), a digital version of a patient's paper chart. Physicians intermittently upload diagnostic reports based on their observations of the EHRs stored in the storage center. Figure 1 shows an example of an mHealth monitoring and data transfer system. Reportedly, a growing number of healthcare-specific mobile applications are available, and it has been estimated that about 500 million patients around the globe will be in the reach of such apps as of 2015 [3].

EHRs contain sensitive information such as patients' medical history, diagnoses, immunization dates, allergies, and medications, which are bound to the real identities of patients. That is, whoever can freely access the storage center is able to learn both the identity and clinical information of a specific patient, which clearly threatens the patient's privacy. Thus, privacy concerns are arguably a major issue, and related requirements are enacted nationwide. For example, in the United States compliance to HIPPA (Health Information Technology for Economic and Clinical Health Act) encourages healthcare providers to not only adopt EHRs but also keep them confidential [4]. This clearly indicates that EHRs must be kept under strict conditions and be accessible only by the authorized user.

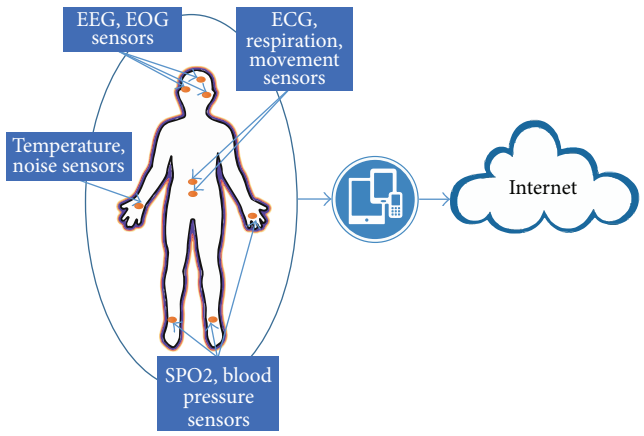


Figure 1: Typical system architecture of mHealth monitoring systems.

Unfortunately, standard encryption schemes are not suitable for mHealth systems for the following reasons [5].

(i) *Absence of Proper Access Control*. Well-known encryption schemes, such as AES, guarantee the confidentiality of data if security parameters are well-chosen. However, such schemes are not designed to support fine-grained access control.

(ii) *Expensive Key Management*. Public key encryption schemes do not support one-to-many relationships between the ciphertext and decryption key, necessitating the burdensome distribution and management of public keys.

Since healthcare delivery is a decentralized process taking place across many institutional boundaries, standard approaches to securing health records include role-based access control because the flexible assignment of permissions to a wide range of user is possible only with fine-grained access control. At the same time, the confidentiality of EHRs must be maintained without hindering clinical care by denying legitimate access requests of authorized users, such as doctors, nurses, lab technicians, researchers, and receptionists [6, 7]. Thus, a variety of policy-based encryption schemes have been proposed to share data securely and provide reliable access control [8–11]. These schemes are promising in that the accessibility of shared data is dependent on the user's capacity to satisfy a given policy. Furthermore, encryptors do not require a priori knowledge of the recipients, such as identities or certificates. Specifically, ciphertext policy attribute-based encryption (CP-ABE) allows the construction of policies by utilizing attributes as public keys, thereby protecting shared data against unauthorized users [12–16]. As access to EHRs varies across the space of uneven distributions of healthcare providers and consumers and among population groups with different socioeconomic and demographic characteristics [17], CP-ABE is a convincing alternative to the conventional cryptographic primitive for mHealth. CP-ABE can provide fine-grained and flexible access control to the shared data in mHealth systems.

It is notable that not only the data, but also the policies for sharing that data are sensitive. Typically, the access policies may reveal sensitive information, such as the underlying data, the identity of a patient, or symptoms indicating what diseases a patient is suffering from. To some extent, patients are reluctant to expose such private information, preferring instead to keep their privacy intact through securing both the EHRs and their access policies. Although CP-ABE provides a desirable access policy, it has one drawback: the access policies attached to ciphertexts are public. From these access policies, unauthorized users can learn information about the underlying data itself. This weakness is known as the policy privacy problem.

To overcome the policy privacy problem, several CP-ABE schemes with hidden access policies were proposed [9, 18]. In these schemes, the encryptor-chosen access policies are associated with each ciphertext in a way hidden such that even an authorized user learns no information about the underlying policy other than that he is authorized to decrypt. Although these schemes feature hidden policies, they suffer from being inefficient; that is, the ciphertext size is linear with respect to the number of attributes in the access policy.

To limit ciphertext size, Zhou et al. introduced a CP-ABE scheme which provides both a hidden access policy and a constant-sized ciphertext [19]. However, their scheme lacks user traceability. In general, most CP-ABE schemes supporting constant-sized ciphertext or hidden access policies cannot trace malicious users who illegally share their decryption keys. Specifically, the secret keys of policy-based encryption consist of sharable attributes so that the decryption keys have no uniquely identifiable information. Thus, if a malicious user leaks his decryption key to others, then there is no clear evidence indicating that the key belongs to him. Although Li et al. proposed a CP-ABE scheme featuring a hidden access policy and traceability [20], it lacks constant-sized ciphertext, resulting in increased communication and storage costs.

Contribution. In this paper, we propose an efficient attribute-based secure data sharing scheme for mHealth with hidden policies and traceability. The proposed scheme enforces hidden access policies with wildcards and supports constant-sized ciphertext, regardless of the number of attributes. Also, we embed a uniquely identifiable point into each decryption key in order to prevent the user from intentionally distributing the decryption key to others, thereby achieving traceability. Additionally, the proposed scheme allows users to outsource part of the decryption process to the more powerful storage center to minimize computation cost at the user side. Our performance results show that the storage center computes almost 50% of the decryption process on behalf of users. To the best of our knowledge, this is the first construction that achieves all these functionalities simultaneously.

Organization. The rest of this paper is organized as follows. We begin with a discussion of related work in Section 2. In Section 3, we describe the cryptographic background and define a general CP-ABE with a hidden policy, constant-sized ciphertext, and traceability. Section 4 describes the mHealth

architecture and security model. In Section 5, we present the construction of the proposed scheme in detail, followed by a performance analysis in Section 6. We analyze its security in Section 7 and conclude the paper in Section 8.

2. Related Work

The idea of Identity-Based Encryption (IBE) was first introduced by Shamir [21]. In IBE, the encryptor makes an access policy based on an identity, and only a user with the matching identity obtains the decryption privilege. Encryption by identity, however, leads to the following limitations: lack of one-to-many relationship between the ciphertext and decryption key and the need for the encryptor to know each user's identity in advance. Later, Sahai and Waters introduced Fuzzy Identity-Based Encryption, which is the first prototype of attribute-based encryption (ABE) [22]. While the IBE scheme views an identity as a string of characters, in ABE, an identity is viewed as a set of descriptive attributes (a.k.a., identity set) such as name and affiliation. The ABE scheme allows the encryption of a message based on some identity set ω', and the decryption ability is given if and only if a user's set ω is close enough to ω' to satisfy a system-defined threshold. This property enables fine-grained access control and a one-to-many relationship between a ciphertext and its receivers since anyone whose identity set satisfies a given threshold can obtain the decryption privilege. However, the threshold semantics are not very expressive and cannot support fine-grained access control. This drawback means that the threshold-based ABE scheme cannot be applied to more general systems.

In CP-ABE [12–16], a ciphertext is associated with an access policy and decryption keys are labeled with an arbitrary number of attributes. The encryptor specifies an access policy over encryptor-chosen attributes. The access right is given if and only if the attributes in the decryption key satisfy the access policy in the ciphertext. In these schemes, however, the size of a ciphertext has a linear relationship with the number of attributes in the access policies, resulting in inapplicability for resource-constrained environments.

To limit the size of ciphertexts, Zhou and Huang proposed constant-sized CP-ABE (C-CP-ABE) with a logical *AND* access policy with wildcards [23]. This scheme limits the size of each ciphertext to up to 300 bytes in total, where a ciphertext consists of encrypted data, an access policy, and 2 bilinear group elements. Chen et al. further improved the C-CP-ABE scheme in terms of security [24] making it CPA-secure under a well-established assumption in the standard model without loss of efficiency. Overall, these schemes successfully make the size of ciphertexts constant. However, they reveal the underlying access policy publicly.

While previous works feature open access policies, Hur introduced a CP-ABE scheme with hidden access policy in smart grid [9]. To preserve policy privacy, a one-way anonymous key agreement scheme is used as a building block in order to replace identity hashes with user-generated pseudonyms. However, this scheme does not support constant-sized ciphertext. Interestingly, an efficient CP-ABE scheme with a hidden policy was proposed [19]. In this scheme, AND-gate access policies with wildcards are used and each ciphertext header requires 2 bilinear group elements, each of which is limited to 100 bytes in total. Also, access policies are obfuscated by computing the intersection between a given access policy and an all-wildcard attribute set. This technique, however, partially leaks the access policy, because unauthorized users can guess at a minimum which attributes are treated as *do not care*. In addition, the user must run the decryption algorithm at least once, to determine whether he satisfies the access policy, since only decryption failure notifies whether the decryption key satisfies the underlying access policy.

The ability to resist illegal key sharing is a highly desirable characteristic for ABE. To achieve this, Li et al. introduced a user-accountable CP-ABE scheme that binds user identity in the private key, thereby allowing illegally-shared keys to be traced [25]. Although this methodology has also been adopted by other traceable CP-ABE schemes [26, 27], none of them fully support either constant-sized ciphertext or hidden access policies. In addition to supporting these features, in this paper, we also insert a unique identifier into each private key such that any key can be traced in constant time, regardless of the number of attributes.

3. Preliminaries

3.1. Bilinear Map. Let $\mathbb{G}_0$ be a multiplicative cyclic group of large prime order p. The bilinear map e is defined as follows: $e : \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_1$, where $\mathbb{G}_1$ is the codomain of e. The bilinear map e has the following properties:

(i) *Bilinearty.* $e(P^a, Q^b) = e(P, Q)^{ab}$, where $\forall P, Q \in \mathbb{G}_0$, $\forall a, b \in \mathbb{Z}_p^*$.

(ii) *Symmetry.* One has $\forall P, Q \in \mathbb{G}_0$, $e(P, Q) = e(Q, P)$.

(iii) *Nondegeneracy.* $e(g, g) \neq 1$, where g is the generator of $\mathbb{G}_0$.

(iv) *Computability.* There exists an efficient algorithm to compute the bilinear map e.

3.2. Security Assumption. The security of the proposed scheme is based on the Bilinear Diffie-Hellman Exponent assumption (BDHE) [28]. Let $\mathbb{G}_0$ be a bilinear group of large prime order p and let g be a generator of $\mathbb{G}_0$. The K-BDHE problem in $\mathbb{G}_0$ is defined as follows. Given the vector of $2K+1$ elements

$$\left(h, g, g^{\alpha}, g^{\alpha^2}, \ldots, g^{\alpha^K}, g^{\alpha^{K+2}}, \ldots, g^{\alpha^{2K}}\right) \in \mathbb{G}_0^{2K+1} \quad (1)$$

as the input where $g^{\alpha^{K+1}}$ is not in the vector, the goal of the computational K-BDHE problem is to compute $e(g, h)^{\alpha^{K+1}}$. Define the set $Y_{g,\alpha,K}$ as

$$Y_{g,\alpha,K} = \left\{g^{\alpha}, g^{\alpha^2}, \ldots, g^{\alpha^K}, g^{\alpha^{K+2}}, \ldots, g^{\alpha^{2K}}\right\}. \quad (2)$$

Then, we have the following definition.

Definition 1 (Decisional K-BDHE). The decisional K-BDHE assumption is said to be hold in $\mathbb{G}_0$ if there is no probabilistic polynomial time adversary who is able to distinguish

$$\left\langle h, g, Y_{g,\alpha,K}, e(g,h)^{\alpha^{(K+1)}} \right\rangle, \quad \left\langle h, g, Y_{g,\alpha,K}, e(g,h)^{R} \right\rangle \tag{3}$$

with nonnegligible advantage, where $\alpha, R \in \mathbb{Z}_p$ and $g, h \in \mathbb{G}_0$ are chosen independently and uniformly at random.

We exploit Boneh et al.'s l-Strong Diffie-Hellman assumption (l-SDH) to prove traceability [29]. Given a $(l+1)$-tuple $(g, g^x, g^{x^2}, \ldots, g^{x^l})$ as input where $x \in \mathbb{Z}_p^*$ is chosen uniformly at random, the l-SDH assumption is stated as follows: there is no probabilistic polynomial time adversary who is able to output $(c, g^{1/(x+c)}) \in \mathbb{Z}_p^* \times \mathbb{G}_0$ with nonnegligible probability, where c is not allowed to be zero.

Formally, we have the following l-SDH assumption.

Assumption 2 (l-SDH). The l-Strong Diffie-Hellman problem in $\mathbb{G}_0$ is defined as follows: given a $(l+1)$-tuple $(g, g^x, g^{x^2}, \ldots, g^{x^l})$ as input, output $(c, g^{1/(x+c)}) \in \mathbb{Z}_p^* \times \mathbb{G}_0$. An algorithm $\mathcal{A}$ has advantage ϵ in solving l-SDH in $\mathbb{G}_0$ if the following holds:

$$\Pr\left[\mathcal{A}\left(g, g^x, g^{x^2}, \ldots, g^{x^l}\right) = \left(c, g^{1/(x+c)}\right)\right] \geq \epsilon, \tag{4}$$

where the probability is over the random choice of x in $\mathbb{Z}_p^*$.

Definition 3. The l-SDH assumption is (t, ϵ)-secure if no t-time algorithm has advantage at least ϵ in solving the l-SDH problem in $\mathbb{G}_0$.

3.3. Access Policy. Given an attribute universe $U = \{A_1, A_2, \ldots, A_k\}$, each A_i has one of three values $\{A_i^+, A_i^-, A_i^*\}$, where A_i^+ denotes that the user has A_i, A_i^- denotes that the user does not have A_i or A_i is not a proper attribute of this user, and A_i^* denotes a wildcard specifying *do not care*. We define the user's attribute set as follows.

Definition 4. Let $L = \{A_1^{+,-}, A_2^{+,-}, \ldots, A_k^{+,-}\}$ be a user's attribute set, where $A_i^{+,-} \in \{A_i^+, A_i^-\}$ and k is the order of the attribute universe. Then, $L = L^+ \cup L^-$, where $L^+ = \{A_i^+ \mid \forall i \in \{1, \ldots, k\}\}$ and $L^- = \{A_i^- \mid \forall i \in \{1, \ldots, k\}\}$. One has $L^+ \cap L^- = \emptyset$.

Next we define the *AND*-gate access policy as follows.

Definition 5. Let $W = \{A_1, \ldots, A_k\}$ be an *AND*-gate access policy where $A_i \in \{A_i^+, A_i^-, A_i^*\}$. Denote $L \vDash W$ that the user's attribute set L satisfies W. Then,

$$L \vDash W \Longleftrightarrow W \subset L \cup \{A_1^*, \ldots, A_k^*\}. \tag{5}$$

3.4. One-Way Anonymous Key Agreement. In this paper, the key idea used to obfuscate attributes in the policy starts from Boneh-Franklin Identity-Based Encryption [30]. In their scheme, a private key generator (PKG) takes the role of issuing private keys. It generates a private key $d_i = H(\mathrm{ID}_i)^s \in \mathbb{G}_0$ for each user ID_i using a master secret s, where $H : \{0,1\}^* \to \mathbb{G}_0$ is a cryptographic hash function.

Based on [30], Kate et al. proposed a one-way anonymous key agreement scheme by replacing $H(\mathrm{ID}_i)$ with a pseudonym chosen by each user [31]. This scheme guarantees anonymity for just one receiver when two users engage in it. We give a specific example as follows. Suppose Alice and Bob hold identity ID_A and identity ID_B, respectively, and they are clients of the same key authority which holds a master secret s. Given the private key $d_A = Q_A^s = H(\mathrm{ID}_A)^s$, Alice wants to communicate with Bob, without disclosing her identity.

To achieve this, the key agreement protocol runs as follows:

(1) Alice computes $Q_B = H(\mathrm{ID}_B)$, chooses a random $r_A \in \mathbb{Z}_p^*$, sets a pseudonym $P_A = Q_A^{r_A}$, and computes the session key $K_{A,B} = e(d_A, Q_B)^{r_A} = e(Q_A, Q_B)^{sr_A}$. She sends the pseudonym P_A to Bob.

(2) Given his private key d_B, Bob computes the session key $K_{A,B} = e(P_A, d_B) = e(Q_A, Q_B)^{sr_A}$.

In this noninteractive manner, the session key is implicitly authenticated such that Alice is assured that the no one can derive the key other than Bob. Based on the BDH assumption, this protocol is proved to be secure in the random oracle model satisfying unconditional anonymity, no impersonation, and session key secrecy. To hide the policy we exploit the technique used in [9] as a building block instead of building a new method for policy obfuscation from scratch.

3.5. Definitions. In this section, we define a general CP-ABE with hidden policy, constant-sized ciphertexts, and traceability capabilities for secure data sharing. The scheme consists of the following seven algorithms:

(i) $Setup(k) \to (MK, PK)$. The Setup algorithm takes as input the number of attributes k. It outputs a public key PK and a master key MK and initializes an identity table $T = \emptyset$.

(ii) $KeyGen(MK, PK, L, id) \to (SK)$. The key generation algorithm takes as input the master key MK, the public key PK, and the user's attribute set L with identity id. It outputs a decryption key SK and inserts id into T.

(iii) $Encrypt(PK, W, M) \to (CT)$. The encryption algorithm takes as input the public key PK, an access policy W, and a message M. It outputs a ciphertext CT such that only the users whose decryption keys satisfying W should be able to extract M. CT is associated with the obfuscated policy W.

(iv) $GenToken(SK_u, \Lambda) \to (TK_{\Lambda,u})$. The token generation algorithm takes as input the user u's secret key SK_u and a set of attributes $\Lambda \vDash W$. It outputs a token $\mathrm{TK}_{\Lambda,u}$.

(v) $PDecrypt(TK_{\Lambda,u}, CT) \to (CT')$. The partial decryption algorithm takes as input the token and outputs a partially decrypted ciphertext CT' for a user u.

(vi) *Decrypt* $(PK, SK, CT', CT) \rightarrow M$ *or* $\perp$. The decryption algorithm takes as input the public key PK, a decryption key SK, and ciphertexts CT′, CT. If $L \vDash W$, then it outputs a message M, where L is the user's attribute set and W is the access policy. Otherwise, it outputs $\perp$ which indicates the failure of decryption.

(vii) *Trace* $(PK, SK, T) \rightarrow id$ *or* $\top$. The tracing algorithm takes as input the public key PK, a decryption key SK, and the table T. It determines whether SK is *well-formed* indicating that SK is the real output of KeyGen. If SK is well-formed, the algorithm outputs an identity id which corresponds to SK. Otherwise it outputs $\top$ implying that SK is not well-formed. The *well-formed* decryption key is guaranteed to work correctly in the well-formed decryption process.

In the proposed scheme, each public key component is mapped to an attribute value A_i. When encrypting data, the encryptor specifies an access policy W, where $A_i \in \{+, -, *\}$. The decryption succeeds only when the user's attribute set L satisfies the (obfuscated) policy W.

4. mHealth Architecture

4.1. System Model. In mHealth systems, intelligent wireless sensors perform data acquisition and processing [32]. Individual sensors monitor certain physiological signals and communicate with each other and the personal server such as a tablet PC as shown in Figure 1. Then, the personal server integrates the data received from the different sensors and plays the role of a gateway by sending data to the upper layer of the mHealth system. From a security point of view, the mHealth system components are categorized as follows:

(1) *Trust Authority*. This is a key entity that issues the public and secret parameters for the mHealth system. It publishes diverse access privileges to individual entities based on their attributes. The trust authority is assumed to be fully trusted in the mHealth system [10].

(2) *Storage Center*. This is a data repository center that stores EHRs. In mHealth systems, hospitals or clinics with certain qualifications certified by the trust authority can be employed as a storage center. It is assumed to be honest-but-curious [10]. Thus, it will honestly execute the assigned tasks and like to learn as much information from the encrypted data as possible.

(3) *Encryptor*. This is a patient who generates data and sends it to the storage center. It uses mobile devices to interact with the storage center. Encryptors are responsible for defining access policy based on attributes, obfuscating the policy, associating it with the data, and encrypting the data according to the policy. Hereafter, we will use "encryptor" and "patient" interchangeably.

(4) *User*. This includes entities such as the patient, physicians, nurses, lab technicians, researchers, or receptionists who want to access EHRs contained in the storage center. A user will be authorized to decrypt a ciphertext given by the storage center if and only if his key satisfies the access policy of that ciphertext.

4.2. Security Model

CPA Security. The security model of the proposed scheme is similar to that of the CP-ABE scheme with constant-sized ciphertexts [23] except that each key query is labeled with an explicit identity and attributes are obfuscated. We first introduce the semantic security game. A CP-ABE scheme is considered to be CPA-secure if no probabilistic polynomial time adversaries have nonnegligible advantages in the following CPA security game.

(i) *Init*. The adversary chooses a challenge access policy W and gives it to the challenger.

(ii) *Setup*. The challenger runs the Setup algorithm and gives the adversary the public parameter PK.

(iii) *Phase 1*. The adversary queries the challenger for decryption keys corresponding to (id, L), where $L \nvDash W$. The challenger answers with a decryption key SK for L. The adversary repeats this phase adaptively.

(iv) *Challenge*. The challenger obtains $\{\langle C_0, C_1\rangle, \mathrm{Key}\}$ by running the Encrypt algorithm. The challenger sets $\mathrm{Key}_0 = \mathrm{Key}$ and picks a random Key_1 of the same length as Key_0. It then flips a random coin $\beta \in \{0, 1\}$ and gives $\{\langle C_0, C_1\rangle, \mathrm{Key}_\beta\}$ to the adversary.

(v) *Phase 2*. It is the same as Phase 1.

(vi) *Guess*. The adversary outputs a guess $\beta' \in \{0, 1\}$.

The adversary wins the game if $\beta' = \beta$ under the restriction that L cannot satisfy the access policy W. The adversary may run Phase 2 to make multiple key queries in the midst of the challenge. Note that the adversary declares the access policy at the start of the game.

The advantage of an adversary in this game is defined as

$$\left|\Pr\left[\beta' = \beta\right] - \frac{1}{2}\right|. \tag{6}$$

Traceability. The traceability definition for the proposed scheme is described by the following security game:

(i) *Setup*. The challenger runs the Setup algorithm to obtain the public parameter PK. Then, the challenger gives PK to the adversary.

(ii) *KeyQuery*. The adversary makes decryption key queries q-times to the challenger, where sets of attributes $(\mathrm{id}_1, L_1), \ldots, (\mathrm{id}_q, L_q)$ correspond to decryption keys.

(iii) *KeyForgery*. The adversary outputs a decryption key SK_*.

The adversary wins the game if the following holds:

(1) $\mathrm{Trace}(\mathrm{PK}, \mathrm{SK}_*, T) \neq \perp$.

(2) $\mathrm{Trace}(\mathrm{PK}, \mathrm{SK}_*, T) \notin \{\mathrm{id}_1, \ldots, \mathrm{id}_q\}$.

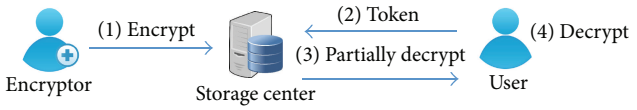


Figure 2: Overview of the proposed data sharing process.

Then, the advantage of the adversary in this game is

$$\Pr\left[\text{Trace}\left(\text{PK}, \text{SK}_*, T\right) \notin \{\perp\} \cup \left\{\text{id}_1, \ldots, \text{id}_q\right\}\right]. \quad (7)$$

Definition 6. A traceable ciphertext policy attribute-based encryption scheme is fully traceable if all polynomial time adversaries have at most negligible advantage in this game.

Policy Privacy. While sharing data in the mHealth system, the storage center or unauthorized users must learn no information about the attributes associated with the access policy of the encrypted data. Also, even authorized users should not obtain any information about these attributes other than the fact that they are authorized to access the data.

5. Proposed Scheme

5.1. System Architecture. The proposed data sharing process in the mHealth system runs as follows. An encryptor defines the access policy with a set of attributes, encrypts the EHRs associated with clinical reports under the policy, and uploads the ciphertext and the obfuscated policy to the storage center. When a user wants to access the uploaded data, he first generates a token using his attributes and sends it to the storage center. If the attributes in the token satisfy the access policy, then the storage center partially decrypts the ciphertext and sends the result to the user. Then, the user finishes the decryption of the ciphertext using his secret key and the partially decrypted ciphertext as inputs. The outline of data sharing process is depicted in Figure 2.

5.2. Scheme Construction. The proposed scheme is constructed on the basis of the following seven algorithms as follows.

Setup $(k) \rightarrow (MK, PK)$. Given k attributes $\{A_1, A_2, \ldots, A_k\}$ as the attribute universe, the proposed scheme has $K = 3k$ attribute values such that $A_i \in \{A_i^+, A_i^-, A_i^*\}$. Specifically, we map $\{A_1^+, A_2^+, \ldots, A_k^+\}$ to $\{1, 2, \ldots, k\}$, $\{A_1^-, A_2^-, \ldots, A_k^-\}$ to $\{k+1, k+2, \ldots, 2k\}$, and $\{A_1^*, A_2^*, \ldots, A_k^*\}$ to $\{2k+1, 2k+2, \ldots, 3k\}$.

Let $\mathbb{G}_0$ be a bilinear group of prime order p. The Setup algorithm chooses a random generator $g \in \mathbb{G}_0$ and random $\alpha, \beta, \gamma \in \mathbb{Z}_p$. For $i = 1, 2, \ldots, K, K+2, \ldots, 2K$, it computes $g_i = g^{(\alpha^i)}$. Then, it computes $v = g^\gamma$ and $h = g^\beta$. The master and public keys are set to $\text{MK} = (\alpha, \beta, \gamma)$; $\text{PK} = (g, g_1, \ldots, g_K, g_{K+2}, \ldots, g_{2K}, v, h) \in \mathbb{G}_0^{2K+1}$. The algorithm initializes an identity table $T = \emptyset$.

KeyGen $(MK, PK, L_{u_t}, id_{u_t}) \rightarrow (SK_{u_t})$. Assume that each user u_t is tagged with an attribute set $L_{u_t} = L_{u_t}^+ \cup L_{u_t}^-$, where $L_{u_t}^+ \subset \{1, 2, \ldots, k\}$ and $L_{u_t}^- \subset \{k+1, k+2, \ldots, 2k\}$. The KeyGen algorithm randomly chooses $a, c \in \mathbb{Z}_p^*$, $\{r_1, r_2, \ldots, r_k\} \in \mathbb{Z}_p$. Then, it computes $r = \sum_{i=1}^k r_i$, $D'' = g^r$ and $D = g^{r\gamma/(a+c)}$. For all $j \in L_{u_t}$, it computes $D''' = H(j)^\beta$, where H is a hash function $H : \{0,1\}^* \rightarrow \mathbb{G}_0$.

Next, the algorithm computes the following:

(i) For every $i \in L_{u_t}^+$, compute $D_i = g^{\gamma(c\alpha^i + r_{i'})/(a+c))}$, where $i' = i$.

(ii) For every $i \in L_{u_t}^-$, compute $D_i = g^{\gamma(c\alpha^i + r_{i'})/(a+c))}$, where $i' = i - k$.

(iii) For every $i \in L_{u_t}^*$, compute $D_i = g^{\gamma(c\alpha^i + r_{i'})/(a+c))}$, where $i' = i - 2k$.

The decryption key for user u_t is set to

$$\text{SK}_{u_t} = \left(D = g^{r\gamma/(a+c)}, \left\{D_i \mid i \in \left\{L_{u_t}^+, L_{u_t}^-, L_{u_t}^*\right\}\right\}, D' = c, D'' = g^r, D''' = H(j)^\beta, D_a = g^a\right). \quad (8)$$

Note that $1/(a+c)$ is computed modulo p. If $\gcd(a+c, p) \neq 1$ or c is already in T, the algorithm is run repeatedly with another random $c \in \mathbb{Z}_p^*$. Then, it puts a tuple (c, id_{u_t}) into T and uploads $(\text{id}_{u_t}, \{g_i^{D'} \mid \forall i \in L_{u_t}\})$ to the storage center.

Encrypt $(PK, W, M) \rightarrow (CT)$. W is an *AND*-gate access policy with k attributes specified by an encryptor u_b, where each attribute is either positive/negative or wildcard. The algorithm chooses a random $b \in \mathbb{Z}_p^*$ and computes $s_j = e(h^b, H(j)), H_1(s_j)$ for all $j \in W$, where H_1 is a hash function $H : \mathbb{G}_1 \rightarrow \{0,1\}^{\log p}$. Then, the access policy W is obfuscated by replacing each attribute with $H_1(s_j)$.

Next, the algorithm picks a random $t \in \mathbb{Z}_p$ and computes a one-time symmetric key $\text{Key} = e(g_K, g_1)^{kt}$. It encrypts the message M as $\{M\}_{\text{Key}}$ and computes g^t. Then, it computes $(v\prod_{j \in W} g_{K+1-j})^t$. The ciphertext CT is set to

$$\text{CT} = \left(W, \{M\}_{\text{Key}}, C_0 = g^t, C_1 = \left(v\prod_{j \in W} g_{K+1-j}\right)^t, \text{id}_{u_t}, g^b\right). \quad (9)$$

The encryptor uploads CT to the storage center.

GenToken $(SK_{u_t}, \Lambda) \rightarrow (TK_{\Lambda,u_t})$. When a user u_t needs to access the ciphertext of u_b in the storage center with a set of attributes $\Lambda \vDash W$, u_t receives g^b from the storage center and generates the token for Λ as follows. For all $j \in \Lambda$, the algorithm computes $s_j = e(g^b, D_j''') = e(g^b, H(j)^\beta)$. Then, it constructs the token $\text{TK}_{\Lambda,u_t} = \{I_j \mid \forall j \in \Lambda,\ I_j = H_1(s_j)\}$. Each I_j will be used as an index for the obfuscated attribute j. The user u_t sends TK_{Λ,u_t} to the storage center.

PDecrypt $(TK_{\Lambda,u_t}, CT) \rightarrow (CT')$. Given TK_{Λ,u_t} from the user u_t, the storage center checks if each I_j in the token satisfies

Table 1: Comparison of different schemes.

	Enc.	Dec.	Ciphertext length	Assumption
Constant-sized ciphertexts [23]	2ex	$2tp$ + ex	$2\lvert\mathbb{G}_0\rvert + \lvert\mathbb{G}_1\rvert$	n-DBDH
Hidden policy [25]	$(t+2)$ex	$(2t+1)p + t$ex	$(t+1)\lvert\mathbb{G}_0\rvert + \lvert\mathbb{G}_1\rvert$	DBDH
Traceability [26]	$(2t+3)$ex	$(2t+1)p + t$ex	$2(t+1)\lvert\mathbb{G}_0\rvert + \lvert\mathbb{G}_1\rvert$	l-BDHI
Proposed	2ex + tp	$(2t+1)(p+$ ex$)$	$3\lvert\mathbb{G}_0\rvert + \lvert\mathbb{G}_1\rvert$	n-BDHE

the access policy associated with CT. If satisfied, the storage center partially decrypts CT using $(\mathrm{id}_{u_t}, \{g_i^{D'} \mid \forall i \in L_{u_t}\})$ as

$$\begin{aligned} A_i &= e\left(g_i^{D'}, C_1\right) = e\left(g, v\prod_{j\in W} g_{K+1-j}\right)^{\alpha^i t D'} \\ &= e\left(g, g^{\gamma+\sum_{j\in W}\alpha^{K+1-j}}\right)^{\alpha^i t D'} \\ &= e\left(g,g\right)^{\alpha^i t D'\gamma + t D'\sum_{j\in W}\alpha^{K+1-j+i}} \end{aligned} \tag{10}$$

for all $i \in W$. Then, it computes a production of all A_i as $\mathrm{CT}' = \prod_{i\in W} A_i$. The storage center sends CT' to u_t.

Decrypt $(PK, SK_{u_t}, CT', CT) \rightarrow M$ *or* $\perp$. On receipt of the partially decrypted ciphertext CT' from the storage center, the user u_t computes B_i for all $i \in W$ as

$$\begin{aligned} B_i &= e\left(C_0, \left(\prod_{j\in W, j\neq i} g_{K+1-j+i}\right)^{D'} \cdot D_i\right) \\ &= e\left(g,g\right)^{t D'\sum_{j\in W, j\neq i}\alpha^{K+1-j+i} + t\gamma(D'\alpha^i + r_i/(a+c))}. \end{aligned} \tag{11}$$

Then, it computes $B = \prod_{i\in W} B_i$ and divides CT' by B. Using the quotient term CT'/B, the user concludes decryption as follows:

$$\begin{aligned} \frac{\mathrm{CT}'}{B}\cdot e\left(D, C_0\right) &= \frac{\mathrm{CT}'}{B}\cdot e\left(g^{\gamma r/(a+c)}, g^t\right) \\ &= e\left(g,g\right)^{D' k t \alpha^{K+1}}. \end{aligned} \tag{12}$$

Then,

$$\begin{aligned} \left(\frac{\mathrm{CT}'}{B}\cdot e\left(D, C_0\right)\right)^{1/D'} &= e\left(g,g\right)^{kt\alpha^{K+1}} \\ = e\left(g^{\alpha^K}, g^{\alpha}\right)^{kt} &= e\left(g_K, g_1\right)^{kt} = \mathrm{Key}. \end{aligned} \tag{13}$$

The user decrypts $\{M\}_{\mathrm{Key}}$.

Trace $(PK, SK_u, T) \rightarrow id_u$ *or* $\top$. SK_u is called well-formed if it passes the following conditions hold:

$$\begin{aligned} D' &\in \mathbb{Z}_p^*, \\ D, D_i, D'' &\in \mathbb{G}_0^{2K+1}, \\ e\left(D_a \cdot g^{D'}, D\right) &= e\left(v, D''\right) \neq 1. \end{aligned} \tag{14}$$

If SK_u is well-formed, the algorithm searches D' in T. If D' is in T, the algorithm outputs the corresponding id_u, and if not, the algorithm outputs the corresponding $\mathrm{id}_\emptyset$ indicating that the corresponding identity never appears in T. If SK_u is not well-formed, the algorithm outputs $\top$.

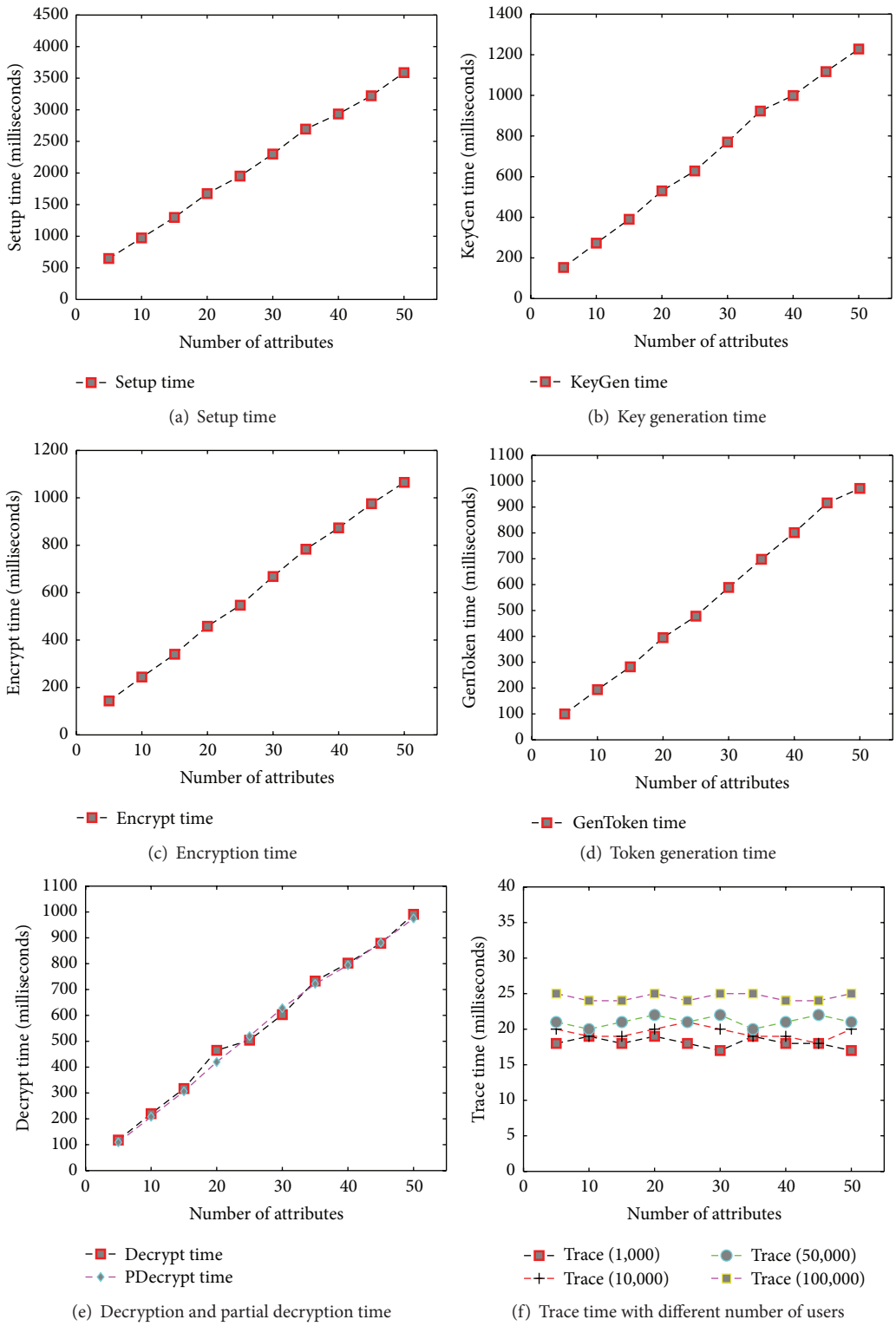
6. Performance Analysis

In this section, we analyze the performance of the proposed scheme compared with the previous schemes including a constant-sized ciphertexts scheme [23], a hidden policy scheme [25], and a traceability scheme [26]. We compare each scheme in several ways such as the computational cost of encryption and decryption and the ciphertext length and in terms of the complexity assumption. Also, we implemented the proposed scheme to evaluate its actual performance. We programmed our system using the Java-based pairing based cryptography (jPBC) library [33] on a GIGABYTE desktop with 4 Intel Core i5-3570 3.40 GHz CPUs, 4 GB RAM, and running Windows 7 Ultimate K.

Table 1 shows the results of comparing the different schemes. The notations we use in the table are as follows: t denotes the number of attributes involved in the access policy, n denotes the number of attributes in the attribute universe, ex denotes the exponentiation operation, and denotes p the paring operation. Note that, following convention, the bit-length of the expression of the access policy and its computational costs over $\mathbb{Z}_p$ are ignored.

In terms of computational cost, the constant-sized ciphertext scheme [23] shows the best encryption phase efficiency, requiring a constant number of exponentiations. The proposed scheme also needs two exponentiations in data encryption, but an additional tp operations are required to obfuscate the access policy. In the decryption phase, the proposed scheme requires more computations than [23] since the user identity is exponentiated to every attribute value to support traceability. In contrast to [25, 26], the proposed scheme requires approximately t number of exponentiations. With regard to the ciphertext length, the proposed scheme and [23] guarantee constant-sized ciphertext. On the other hand, the hidden policy scheme [25] and the traceability scheme [26] incur linearly increasing ciphertexts as the attribute number t increases. Overall, the proposed scheme is efficient in terms of the ciphertext size and provides hidden policy traceability at the cost of more exponentiation operations.

Figure 3 shows the computation overhead incurred in the core algorithms, Setup, KeyGen, Encrypt, GenToken, Decrypt, PDecrypt, and Trace, under various conditions. Figure 3(a) shows how system-wide setup time varies according



(a) Setup time

(b) Key generation time

(c) Encryption time

(d) Token generation time

(e) Decryption and partial decryption time

(f) Trace time with different number of users

Figure 3: Time costs of different algorithms.

to the number of attributes. Figure 3(b) shows the total key generation time against different numbers of attributes. The setup occurs only once at the start of the system, and key generation occurs every time a new user joins. Figure 3(c) shows encryption time against different numbers of attributes. It increases linearly due to the time taken to obfuscate the policy attached to the data. Figure 3(d) shows the token generation time against the number of attributes. The token

Table 2: jPBC and PBC benchmark comparison results [33].

Operation	jPBC	PBC
Pairing	14.654	2.688
Exponentiation in $\mathbb{G}_1$	18.592	4.122
Exponentiation in $\mathbb{G}_T$	2.112	0.529
Exponentiation in $\mathbb{Z}_r$	0.068	0.087

generation process requires a pairing operation time linear to the number of attributes. Figure 3(e) shows the partial decryption time at the storage center and decryption time at the user against the number of attributes. Interestingly, the storage center can undertake nearly 50% of whole decryption process on behalf of users. This property can be most useful for relatively resource-constrained user side devices. Lastly, Figure 3(f) shows the trace time with different numbers of attributes and users. The trace time depends only but not strongly on the number of users.

Further Efficiency Improvement. jPBC is a complete Java port of the PBC library which was originally written in C [34]. Java is widely considered to be slower than C because Java programs run on the Java Virtual Machine rather than directly on the computer's processor. Based on this, we additionally provide benchmark comparison results between jPBC and PBC in order to demonstrate how fast the proposed scheme can be when it is implemented in C language [33]. Table 2 shows the performance comparison between Java and C with respect to pairing and exponentiation operations conducted on the same machine. The two libraries were applied to the curve $y^2 = x^3 + x$ over the field $\mathbb{F}_q$ for some prime $q = 3 \bmod 4$. The order of $\mathbb{F}_q$ is some prime factor of $q + 1$ [33]. Since the cost of the pairing operation in PBC is approximately 12 seconds less than in jPBC, PBC is expected to improve the performance of pairing-dependent algorithms, such as GenToken and policy obfuscation process in Encrypt, by up to 81%. Similarly, the cost of the exponentiation operations in $\mathbb{G}_1$ and $\mathbb{G}_T$ are reduced by 14.47 and 1.583 seconds, respectively. Such a difference between the two libraries implies that moving from Java to C implementation of the proposed scheme can speed up the Setup and KeyGen algorithms by approximately 77.8% and the PDecrypt and Decrypt algorithms by approximately 74.9%.

7. Security Analysis

7.1. Data Confidentiality. In this section, we reduce the chosen plaintext attack (CPA) security of the proposed scheme to a decisional K-BDHE problem. Given an access policy W, a user with an attribute set $L \not\models W$ colludes with $x \leq k$ decryption proxies. Intuitively, this attack works successfully if $L \cup \{i_1, \ldots, i_x\} \models W$. Based on the CPA security game in Section 4.2, we have the following.

Theorem 7. *If a probabilistic polynomial time adversary wins the CPA security game with a nonnegligible advantage, then one can construct a simulator that distinguishes a K-DBHE tuple with a nonnegligible advantage.*

Proof. Suppose that an adversary $\mathcal{A}$'s advantage for winning the game is ϵ. Then, we can construct a simulator $\mathfrak{B}$ which solves the decisional K-BDHE problem with the advantage $\epsilon/2$. The simulator $\mathfrak{B}$ takes an input vector $(h, g, Y_{g,\alpha,K}, Z)$, where Z is either $e(g,h)^{\alpha^{K+1}}$ or a random element in $\mathbb{G}_0$. Then, $\mathfrak{B}$ breaks the decisional K-BDHE problem with the advantage $\epsilon/2$. Specifically, $\mathfrak{B}$ takes a random decisional K-BDHE challenge $\langle h, g, Y_{g,\alpha,K}, Z\rangle$ as input, where Z is either $Z = e(g,h)^{\alpha^{K+1}}$ or a random value.

Next, $\mathfrak{B}$ runs the following CPA game with the role of challenger.

(i) *Init*. $\mathcal{A}$ sends an access policy W to $\mathfrak{B}$.

(ii) *Setup*. $\mathfrak{B}$ runs the Setup algorithm to obtain PK and chooses a random $d \in \mathbb{Z}_p$. Then, $\mathfrak{B}$ computes

$$v = g^d \left(\prod_{j \in W} g_{K+1-j} \right)^{-1} = g^{\gamma}. \tag{15}$$

$\mathfrak{B}$ outputs the public key $\mathrm{PK} = (g, Y_{g,\alpha,K}, v) \in \mathbb{G}_0^{2K+1}$.

Phase 1. The adversary $\mathcal{A}$ submits L, where $L \not\models W$. Then, there exists $j \in L$ such that $j + k \in W$, where $j \in \{1, \ldots, k\}$, or $j - k \in W$, where $j \in \{k+1, \ldots, 2k\}$.

For $i = 1, \ldots, k$, $\mathfrak{B}$ picks k random $r_i \in \mathbb{Z}_p$ and sets $r = r_1 + \cdots + r_k$. Next, $\mathfrak{B}$ randomly chooses $a, c \in \mathbb{Z}_p^*$ and computes

$$D = \left(g^d \prod_{j \in W} \left(g_{K+1-j} \right)^{-1} \right)^{r/(a+c)} = g^{\gamma r/(a+c)}. \tag{16}$$

Next, $\mathfrak{B}$ computes

$$D_i = g_i^d \prod_{j \in W} \left(g_{K+1-j+i} \right)^{-c} \cdot \prod_{j \in W} \left(g_{K+1-j} \right)^{-r_{i'}/(a+c)}, \tag{17}$$

where i falls into one of the following conditions: (1) $i + k \in W$ for all $i \in L^+$, (2) $i - k \in W$ for all $i \in L^-$, and (3) $i \notin W$ for all $i \in L^*$.

Then, each D_i is valid such that

$$\begin{aligned} D_i &= \left(g^d \left(\prod_{j \in W} g_{K+1-j} \right)^{-1} \right)^{c\alpha^i + r_{i'}/(a+c)} \\ &= g^{\gamma(c\alpha^i + r_{i'}/(a+c))}. \end{aligned} \tag{18}$$

Challenge. $\mathfrak{B}$ sets $C_0 = h$ and $C_1 = h^d$ and gives the challenge $\langle C_0, C_1, Z^k\rangle$ to $\mathcal{A}$. Note that $C_0 = h = g^t$ for some t such that

$$\begin{aligned} h^d = \left(g^d\right)^t &= \left(g^d \prod_{j \in W} \left(g_{K+1-j} \right)^{-1} \cdot \prod_{j \in W} \left(g_{K+1-j} \right) \right)^t \\ &= \left(v \prod_{j \in W} \left(g_{K+1-j} \right) \right)^t, \end{aligned} \tag{19}$$

and $Z^k = \mathrm{Key}$ if $Z = e(g,h)^{\alpha^{(K+1)}}$.

Phase 2. Repeat Phase 1.

Guess. The adversary $\mathcal{A}$ outputs a guess b', where $b' = 0$ implies that $Z = e(g,h)^{\alpha^{K+1}}$. If $b' = 1$, then Z is a random element which indicates that $\Pr[\mathfrak{B}(h, g, Y_{g,\alpha,K}, Z) = 0] = 1/2$. Note that each decryption proxy $p_i(r)$ simulates a legal decryption key component with a random r. Specifically, the adversary $\mathcal{A}$ passes r as a guess of $r_{i'}$ which is embedded in D_i, where $i \in W$. We further define a decryption proxy to model collusion attacks. □

Definition 8. Given $2k$ decryption proxies in the security game, each decryption proxy $p_i(r) = g^{\gamma(c\alpha^i + r/(a+c))}$, where $r \in \mathbb{Z}_p$ and $i \in \{1, \ldots, 2k\}$.

Lemma 9 (collision with 1 decryption proxy). *Suppose that $\mathcal{A}$ has issued q queries and there is only 1 attribute $i \notin W$, where $\mathcal{A}$ makes l queries to $p_i(r)$. The probability that none of the queries returns a legal decryption key component of any q is $(1 - q/p)^l$.*

Proof. The probability that at least one query returns an illegal decryption key component of any q is $1 - q/p$. Thus, if none of the l queries succeeds, then $\Pr[r \neq r_{i'}] = (1 - q/p)^l$, where r is a random number in the decryption proxy and $r_{i'}$ is a random number in the decryption key. □

Lemma 10 (collision with multiple decryption proxies). *Suppose $\mathcal{A}$ has issued q queries and there are m attributes dissatisfying W, where $\mathcal{A}$ makes l queries to each decryption proxy $p_{i_1}(r_1), p_{i_2}(r_2), \ldots, p_{i_m}(r_m)$. The probability that none of the queries returns a legal decryption key component of any q is $(1 - (1 - q/p)^l)^m$.*

Proof. The probability that one decryption proxy fails is $\Pr[r \neq r_{i'}] = (1 - q/p)^l$. Thus, the probability that all m decryption proxies succeed is $(1 - (1 - q/p)^l)^m$. □

In case of $Z = e(g,h)^{\alpha^{(K+1)}}$, we have 3 collusion scenarios as follows.

0-Collusion. If no decryption proxy is used, then $\mathcal{A}$ has at least $\epsilon/2$ advantage in breaking the proposed scheme. Thus, $\mathfrak{B}$ has at least the following advantage in breaking K-BDHE problem:

$$\left| \Pr\left[\mathfrak{B}\left(h, g, Y_{g,\alpha,K}, Z\right) = 0\right] - \frac{1}{2} \right| \geq \frac{\epsilon}{2}. \tag{20}$$

1-Collusion. If one decryption proxy $p_i(r)$ is used, then we have $\Pr[r \neq r_{i'}] = (1 - q/p)^l$. Thus, if $\mathcal{A}$ has at least ϵ advantage in breaking the proposed scheme, then $\mathfrak{B}$ has at least $(1 - q/p)^l \epsilon/2$ advantage in breaking the K-BDHE problem.

m-Collusion. If m decryption proxies $p_{i_1}(r_1), \ldots, p_{i_m}(r_m)$ are used, then we have

$$\Pr\left[r_{i_j} \neq r_{i'_j},\ \exists j \leq m\right] = \left(1 - \left(1 - \frac{q}{p}\right)^l\right)^m. \tag{21}$$

Thus, if $\mathcal{A}$ has at least ϵ advantage in breaking the proposed scheme, then $\mathfrak{B}$ has at least the following advantage in breaking the K-BDHE problem:

$$\left(1 - \left(1 - \left(1 - \frac{q}{p}\right)^l\right)^m\right) \cdot \frac{\epsilon}{2}. \tag{22}$$

7.2. Traceability. In this section, we prove the traceability of the proposed scheme based on the l-SDH assumption.

Theorem 11. *If l-SDH assumption holds, then the proposed scheme is fully traceable provided that $q < l$.*

Proof. Suppose that there is a PPT adversary $\mathcal{A}$ who wins the traceability game with nonnegligible advantage ϵ after q key queries. Without loss of generality, assume that $l = q + 1$. Then, we can construct a PPT simulator $\mathfrak{B}$ that breaks l-SDH assumption with nonnegligible advantage.

$\mathfrak{B}$ is given an instance of the l-SDH problem as follows. Let $\mathbb{G}_0$ be a bilinear group of prime order p, let $\overline{g} \in \mathbb{G}_0$, let $e : \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_1$ be a bilinear map, and let $a \in \mathbb{Z}_p$. $\mathfrak{B}$ is given $\mathrm{IN}_{\mathrm{SDH}} = (p, \mathbb{G}_0, \mathbb{G}_1, e, \overline{g}, \overline{g}^a, \ldots, \overline{g}^{a^l})$ as in instance of the l-SDH problem. $\mathfrak{B}$'s goal is to output a pair $(c_r, w_r) \in \mathbb{Z}_p^* \times \mathbb{G}_0$ satisfying $w_r = \overline{g}^{1/(a+c_r)}$ for solving the l-SDH problem. $\mathfrak{B}$ sets $A_i = \overline{g}^{a^i}$ for $i = 0, 1, \ldots, l$ and interacts with $\mathcal{A}$ in the traceability game as follows.

Setup. $\mathfrak{B}$ randomly picks q distinct values $c_1, \ldots, c_q \in \mathbb{Z}_p^*$. Let $f(y)$ be the polynomial $f(y) = \prod_{i=1}^{q}(y + c_i)$. Expand $f(y)$ and write $f(y) = \sum_{i=0}^{q} \alpha_i y^i$, where $\alpha_0, \alpha_1, \ldots, \alpha_q \in \mathbb{Z}_p$ are the coefficients of the polynomial $f(y)$. $\mathfrak{B}$ computes

$$\begin{aligned} g &\longleftarrow \prod_{i=0}^{q} (A_i)^{\alpha_i} = \overline{g}^{f(a)}, \\ g^a &\longleftarrow \prod_{i=1}^{q+1} (A_i)^{\alpha_{i-1}} = \overline{g}^{f(a)\cdot a}. \end{aligned} \tag{23}$$

$\mathfrak{B}$ randomly chooses $\alpha, \gamma \in \mathbb{Z}_p$ and computes $v = g^\gamma$. For $i = 1, 2, \ldots, K, K+2, \ldots, 2K$, $\mathfrak{B}$ sets $g_i = g^{(\alpha^i)}$, where $K = 3k = l$. $\mathfrak{B}$ then gives $\mathcal{A}$ the public parameter

$$\mathrm{PK} = (g, g_1, \ldots, g_K, g_{K+2}, \ldots, g_{2K}, v) \in \mathbb{G}_0^{2K+1}. \tag{24}$$

KeyQuery. $\mathcal{A}$ submits (id_x, L) to $\mathfrak{B}$ to request a decryption key. Assume that it is the xth query. For $x \leq q$, let $f_x(y)$ be the polynomial $f_x(y) = f(y)/(y + c_x) = \prod_{j=1, j\neq x}^{q}(y + c_j)$. Expand $f_x(y)$ and write $f_x(y) = \sum_{j=0}^{q-1} \beta_j y^j$, where $\beta_0, \beta_1, \ldots, \beta_{q-1} \in \mathbb{Z}_p$. $\mathfrak{B}$ computes

$$\sigma_x \longleftarrow \prod_{j=0}^{q-1} \left(A_j\right)^{\beta_j} = \overline{g}^{f_x(a)} = \overline{g}^{f(a)/(a+c_x)} = g^{1/(a+c_x)}. \tag{25}$$

$\mathfrak{B}$ randomly chooses $\{r_1, r_2, \ldots, r_k\} \in \mathbb{Z}_p$. Then, it computes $r = \sum_{i=1}^{k} r_i$, $D'' = g^r$, and $D = \sigma_x^{r\gamma}$.

Finally, $\mathfrak{B}$ computes the following:

(i) For every $i \in L_u^+$, compute $D_i = g^{\gamma c_x \alpha^i} \sigma_x^{\gamma r_{i'}}$, where $i' = i$.

(ii) For every $i \in L_u^-$, compute $D_i = g^{\gamma c_x \alpha^i} \sigma_x^{\gamma r_{i'}}$, where $i' = i - k$.

(iii) For every $i \in L_u^*$, compute $D_i = g^{\gamma c_x \alpha^i} \sigma_x^{\gamma r_{i'}}$, where $i' = i - 2k$.

$\mathfrak{B}$ responds to $\mathscr{A}$ with $\mathrm{SK}_{\mathrm{id}_x, L_x}$ as

$$\mathrm{SK}_{\mathrm{id}_x, L_x} = \left(D = \sigma_x^{r\gamma}, \{D_i \mid i \in \{L_u^+, L_u^-, L_u^*\}\}, D' = c_x, D'' = g^r\right). \tag{26}$$

$\mathfrak{B}$ puts tuple (c_x, id_x) into T.

KeyForgery. $\mathscr{A}$ submits to $\mathfrak{B}$ a decryption key SK_*.

Note that the distributions of PK and SK in the above game are the same as in the real game. Let $\Upsilon_{\mathscr{A}}$ denote the event that $\mathscr{A}$ wins the game; that is, SK_* is well-formed, and $c_r \notin \{c_1, c_2, \ldots, c_q\}$. The adversary's advantage over the game is $\epsilon/2$ since there is no decryption proxy used. If $\Upsilon_{\mathscr{A}}$ does not happen, $\mathfrak{B}$ chooses a random a random pair $(c_r, w_r) \in \mathbb{Z}_p^* \times \mathbb{G}_0$ as its solution for l-SDH problem. If $\Upsilon_{\mathscr{A}}$ happens, $\mathfrak{B}$ writes the polynomial $f(y) = \gamma(y)(y + D') + \gamma_{-1}$ for some polynomial $\gamma(y) = \sum_{i=0}^{q-1} (\gamma_i y^i)$ and some $\gamma_{-1} \in \mathbb{Z}_p$. Then, $\gamma_{-1} \neq 0$ since $f(y) = \prod_{i=1}^{q} (y + c_i)$, where $c_i \in \mathbb{Z}_p^*$ and $D' \notin \{c_1, c_2, \ldots, c_q\}$. Thus $y + D'$ does not divide $f(y)$. $\mathfrak{B}$ computes the value of $\gcd(\gamma_{-1}, p)$.

Next, let $\Omega_{\mathrm{SDH}}(c_r, w_r)$ denote the event that (c_r, w_r) is a solution to the l-SDH problem. Note that when $\mathfrak{B}$ chooses (c_r, w_r) randomly, $\Omega_{\mathrm{SDH}}(c_r, w_r)$ happens with negligible probability, say zero. $\mathfrak{B}$ solves the l-SDH problem with probability

$$\begin{aligned}
&\Pr\left[\Omega_{\mathrm{SDH}}(c_r, w_r)\right] \\
&= \Pr\left[\Omega_{\mathrm{SDH}}(c_r, w_r) \mid \overline{\Upsilon_{\mathscr{A}}}\right] \cdot \Pr\left[\overline{\Upsilon_{\mathscr{A}}}\right] \\
&\quad + \Pr\left[\Omega_{\mathrm{SDH}}(c_r, w_r) \mid \Upsilon_{\mathscr{A}} \wedge \gcd(\gamma_{-1}, p) \neq 1\right] \\
&\quad \cdot \Pr\left[\Upsilon_{\mathscr{A}} \wedge \gcd(\gamma_{-1}, p) \neq 1\right] \\
&\quad + \Pr\left[\Omega_{\mathrm{SDH}}(c_r, w_r) \mid \Upsilon_{\mathscr{A}} \wedge \gcd(\gamma_{-1}, p) = 1\right] \\
&\quad \cdot \Pr\left[\Upsilon_{\mathscr{A}} \wedge \gcd(\gamma_{-1}, p) = 1\right] \\
&= 0 + 0 + 1 \cdot \Pr\left[\Upsilon_{\mathscr{A}} \wedge \gcd(\gamma_{-1}, p) = 1\right] \leq \epsilon.
\end{aligned} \tag{27}$$

Thus, $\mathfrak{B}$ can break the l-SDH assumption with advantage $\leq \epsilon$. □

7.3. Policy Privacy. When an encryptor uploads its ciphertext to the storage center, every attribute j in the access policy is obfuscated as $H_1(e(h^b, H(j)))$ with a random b using the one-way anonymous key agreement protocol [31] such that only users in possession of valid corresponding attributes are able to compute the same value. It is infeasible to guess j from $H_1(e(h^b, H(j)))$ without having the corresponding attributes due to b which is chosen uniformly at random by the encryptor. Specifically, the storage center does not have $D''' = H(j)^\beta$ which is a secret key component owned by users whose attribute sets satisfy the access policy. Due to the secrecy property of the key agreement protocol [31], the storage center cannot compute $e(g^b, H(j)^\beta)$.

In token generation phase, a user computes indices $I_j = H_1(e(g^b, H(j)^\beta))$ for each (obfuscated) attribute j. Due to the secrecy property of the key agreement protocol, only the authorized users are able to construct indices corresponding to j. Thus, the storage center cannot generate correct indices for the attributes in the access policy. Also, even though the storage center conducts partial decryptions, the user learns nothing about the underlying access policy except that he can decrypt the ciphertext since he receives only the partially decrypted value and no more. Therefore, the proposed scheme guarantees the policy privacy against the storage center and authorized users.

8. Conclusion

In this paper, we proposed an efficient attribute-based secure mHealth data sharing scheme with hidden policies and traceability. The proposed scheme significantly reduces storage and communication costs. The access policies are obfuscated such that not only data privacy but also policy privacy is preserved. The computational costs of users are reduced by delegating approximately 50% of the decryption operation to the more powerful storage systems. Lastly, the proposed scheme is able to trace malicious users who illegally leak their keys. Our security analysis shows that the proposed scheme is secure against chosen-ciphertext and key forgery attacks under the decisional K-BDHE and l-SDE assumptions. We also prove that the policy privacy of the proposed scheme is preserved against the storage center and authorized users.

Competing Interests

The authors declare that they have no competing interests.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (no. 2016R1A2A2A05005402). This work was also supported by Institute for Information & Communications Technology Promotion (IITP) grant funded by the Korea government (MSIP) (no. B0190-15-2028). This work was also supported by the research fund of Signal Intelligence Research Center supervised by the Defense Acquisition Program Administration and Agency for Defense Development of Korea.

References

[1] P. Germanakos, C. Mourlas, and G. Samaras, "A mobile agent approach for ubiquitous and personalized eHealth information systems," in *Proceedings of the Workshop on 'Personalization for*

e-Health' of the 10th International Conference on User Modeling, pp. 67–70, Edinburgh, UK, July 2005.

[2] E. Jovanov, A. O'Donnell Lords, D. Raskovic, P. G. Cox, R. Adhami, and F. Andrasik, "Stress monitoring using a distributed wireless intelligent sensor system," *IEEE Engineering in Medicine and Biology Magazine*, vol. 22, no. 3, pp. 49–55, 2003.

[3] J. A. Wolf, J. F. Moreau, O. Akilov et al., "Diagnostic inaccuracy of smartphone applications for melanoma detection," *JAMA Dermatology*, vol. 149, no. 4, pp. 422–426, 2013.

[4] United States Department of Health & Human Services, *Health Information Privacy*, 2011, http://www.hhs.gov/ocr/privacy/index.html.

[5] S. Alshehri, S. P. Radziszowski, and R. K. Raj, "Secure access for healthcare data in the cloud using Ciphertext-policy attribute-based encryption," in *Proceedings of the IEEE 28th International Conference on Data Engineering Workshops (ICDEW '12)*, pp. 143–146, IEEE, Arlington, Va, USA, April 2012.

[6] M. Poulymenopoulou, F. Malamateniou, and G. Vassilacopoulos, "E-EPR: a cloud-based architecture of an electronic emergency patient record," in *Proceedings of the 4th ACM International Conference on PErvasive Technologies Related to Assistive Environments (PETRA '11)*, article 35, Crete, Greece, May 2011.

[7] H. A. J. Narayanan and M. H. Gunes, "Ensuring access control in cloud provisioned healthcare systems," in *Proceedings of the IEEE Consumer Communications and Networking Conference (CCNC '11)*, pp. 247–251, Las Vegas, Nev, USA, January 2011.

[8] R. Bobba, H. Khurana, M. Alturki, and F. Ashraf, "PBES: a policy based encryption system with application to data sharing in the power grid," in *Proceedings of the 4th International Symposium on ACM Symposium on Information, Computer and Communications Security (ASIACCS '09)*, pp. 262–275, March 2009.

[9] J. Hur, "Attribute-based secure data sharing with hidden policies in smart grid," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 11, pp. 2171–2180, 2013.

[10] L. Guo, C. Zhang, J. Sun, and Y. Fang, "PAAS: a privacy-preserving attribute-based authentication system for eHealth networks," in *Proceedings of the 32nd IEEE International Conference on Distributed Computing Systems (ICDCS '12)*, pp. 224–233, IEEE, Macau, June 2012.

[11] A. Kapadia, P. P. Tsang, and S. W. Smith, "Attribute-based publishing with hidden credentials and hidden policies," *Proceedings of the 14th Annual Network and Distributed System Security Symposium (NDSS '07)*, vol. 7, pp. 179–192, 2007.

[12] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attributebased encryption," in *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 321–334, Berkeley, Calif, USA, May 2007.

[13] B. Waters, "Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization," in *Public Key Cryptography-PKC*, pp. 53–70, 2011.

[14] V. Goyal, A. Jain, O. Pandey, and A. Sahai, "Bounded ciphertext policy attribute based encryption," in *Automata, Languages and Programming: 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7–11, 2008, Proceedings, Part II*, vol. 5126 of *Lecture Notes in Computer Science*, pp. 579–591, Springer, Berlin, Germany, 2008.

[15] L. Ibraimi, M. Petkovic, S. Nikova, P. Hartel, and W. Jonker, "Mediated ciphertext-policy attribute-based encryption and its application," in *Information Security Applications*, H. Y. Youm and M. Yung, Eds., vol. 5932 of *Lecture Notes in Computer Science*, pp. 309–323, 2009.

[16] T. Jung, X.-Y. Li, Z. Wan, and M. Wan, "Privacy preserving cloud data access with multi-authorities," in *Proceedings of the IEEE INFOCOM*, pp. 2625–2633, Turin, Italy, April 2013.

[17] F. Wang and W. Luo, "Assessing spatial and nonspatial factors for healthcare access: towards an integrated approach to defining health professional shortage areas," *Health & Place*, vol. 11, no. 2, pp. 131–146, 2005.

[18] R. W. Bradshaw, J. E. Holt, and K. E. Seamons, "Concealing complex policies with hidden credentials," in *Proceedings of the 11th ACM Conference on Computer and Communications Security (CCS '04)*, pp. 146–157, October 2004.

[19] Z. Zhou, D. Huang, and Z. Wang, "Efficient privacy-preserving ciphertext-policy attribute based-encryption and broadcast encryption," *IEEE Transactions on Computers*, vol. 64, no. 1, pp. 126–138, 2015.

[20] J. Li, K. Ren, B. Zhu, and Z. Wan, "Privacy-aware attribute-based encryption with user accountability," in *Information Security*, pp. 347–362, Springer, Berlin, Germany, 2009.

[21] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Advances in Cryptology*, G. R. Blakley and D. Chaum, Eds., vol. 196 of *Lecture Notes in Computer Science*, pp. 47–53, 1985.

[22] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Advances in Cryptology—EUROCRYPT 2005*, vol. 3494 of *Lecture Notes in Computer Science*, pp. 457–473, Springer, Berlin, Germany, 2005.

[23] Z. Zhou and D. Huang, "On efficient ciphertext-policy attribute based encryption and broadcast encryption," in *Proceedings of the 17th ACM Conference on Computer and Communications Security (CCS '10)*, pp. 753–755, ACM, Chicago, Ill, USA, October 2010.

[24] C. Chen, Z. Zhang, and D. Feng, "Efficient ciphertext policy attributebased encryption with constant-size ciphertext and constant computationcost," in *Provable Security: 5th International Conference, ProvSec 2011, Xi'an, China, October 16–18, 2011. Proceedings*, vol. 6980 of *Lecture Notes in Computer Science*, pp. 84–101, Springer, Berlin, Germany, 2011.

[25] J. Li, K. Ren, and Z. Wan, "Privacy-aware attribute-based encryption with user accountability," in *Information Security*, pp. 347–362, Springer, Berlin, Germany, 2009.

[26] Z. Liu, Z. Cao, and D. S. Wong, "White-box traceable ciphertext-policy attribute-based encryption supporting any monotone access structures," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 1, pp. 76–88, 2013.

[27] Z. Liu, Z. Cao, and D. S. Wong, "Blackbox traceable CP-ABE: How to catch people leaking their keys by selling decryption devices on eBay," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security (CCS '13)*, pp. 475–486, ACM, November 2013.

[28] D. Boneh, X. Boyen, and E. J. Goh, "Hierarchical identity based encryption with constant size ciphertext," in *Advances in Cryptology—EUROCRYPT 2005*, pp. 440–456, Springer, Berlin, Germany, 2005.

[29] D. Boneh and X. Boyen, "Short signatures without random oracles," in *Advances in Cryptology-EUROCRYPT 2004*, vol. 3027 of *Lecture Notes in Computer Science*, pp. 56–73, Springer, Berlin, Germany, 2004.

[30] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," in *Advances in Cryptology—CRYPTO 2001*, pp. 213–229, Springer, Berlin, Germany, 2001.

[31] A. Kate, G. Zaverucha, and I. Goldberg, "Pairing-based onion routing," in *Privacy Enhancing Technologies*, N. Borisov and P.

Golle, Eds., vol. 4776 of *Lecture Notes in Computer Science*, pp. 95–112, Springer, Berlin, Germany, 2007.

[32] E. Jovanov and D. Raskovic, "Wireless intelligent sensors," in *M-Health*, pp. 33–49, Springer, New York, NY, USA, 2006.

[33] A. De Caro and V. Iovino, "jPBC: java pairing based cryptography," in *Proceedings of the IEEE Symposium on Computers and Communications (ISCC '11)*, pp. 850–855, June-July 2011.

[34] B. Lynn, *The Pairing-Based Cryptography (PBC) Library*, 2010, http://crypto.stanford.edu/pbc.