Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 6682580, 16 pages
https://doi.org/10.1155/2021/6682580

Research Article

Traceable Multiauthority Attribute-Based Encryption with Outsourced Decryption and Hidden Policy for CIoT

Suhui Liu,[1] **Jiguo Yu**,[2,3,4] **Chunqiang Hu**,[5,6] **and Mengmeng Li**[1]

[1]*School of Computer Science, Qufu Normal University, Rizhao, 276826 Shandong, China*
[2]*School of Computer Science and Technology, Qilu University of Technology (Shandong Academy of Sciences), Jinan, Shandong 250353, China*
[3]*Shandong Computer Science Center (National Supercomputer Center in Jinan), Jinan, Shandong 250014, China*
[4]*Shandong Laboratory of Computer Networks, Jinan 250014, China*
[5]*School of Big Data and Software Engineering, Chongqing University, Chongqing 400044, China*
[6]*Key Laboratory of Dependable Service Computing in Cyber Physical Society, Ministry of Education (Chongqing University), China*

Correspondence should be addressed to Jiguo Yu; jiguoyu@sina.com and Chunqiang Hu; chu@cqu.edu.cn

Received 31 October 2020; Revised 1 January 2021; Accepted 7 April 2021; Published 21 April 2021

Academic Editor: Yingjie Wang

Cloud-assisted Internet of Things (IoT) significantly facilitate IoT devices to outsource their data for high efficient management. Unfortunately, some unsettled security issues dramatically impact the popularity of IoT, such as illegal access and key escrow problem. Traditional public-key encryption can be used to guarantees data confidentiality, while it cannot achieve efficient data sharing. The attribute-based encryption (ABE) is the most promising way to ensure data security and to realize one-to-many fine-grained data sharing simultaneously. However, it cannot be well applied in the cloud-assisted IoT due to the complexity of its decryption and the decryption key leakage problem. To prevent the abuse of decryption rights, we propose a multiauthority ABE scheme with white-box traceability in this paper. Moreover, our scheme greatly lightens the overhead on devices by outsourcing the most decryption work to the cloud server. Besides, fully hidden policy is implemented to protect the privacy of the access policy. Our scheme is proved to be selectively secure against replayable chosen ciphertext attack (RCCA) under the random oracle model. Some theory analysis and simulation are described in the end.

1. Introduction

In traditional public key encryption schemes, the encryptor encrypts the message with the public key of the decryptor; hence, only the decryptor who owns the corresponding decryption key can decrypt the data. In other words, this type of scheme relies on the public key certificate system which we all know is pretty difficult to manage. In 1984, Shamir first proposed the identity-based encryption (IBE) where the encryptor uses the identity of the decryptor as his/her public key [1]. In [2], Boneh et al. proposed an IBE using the elliptic curve pairing, which greatly promoted the development of this field. Although the IBE solves the public key management problem, it still cannot achieve one-to-many private data sharing. Unfortunately, this kind of application is extremely common in ubiquitous Internet of Things (IoT) scenarios.

To tackle this issue, Sahai et al. first proposed a fuzzy identity encryption scheme [3], which is later developed into the attribute-based encryption (ABE). There are two types of ABE, the first one is named as ciphertext-policy attribute-based encryption (CP_ABE) and the other one is key-policy attribute-based encryption (KP_ABE). CP_ABE was proposed by Waters, in which the encryptor needs to know nothing about who can decrypt the ciphertext exactly, and he/she just encrypts the message with a self-defined access policy [4]. Any decryptor can decrypt correctly as long as its attribute set meets the access policy in the ciphertext. In other words, in CP_ABE schemes, data owners own the right to design who can decrypt fully.

IoT, which acts as the bridge between the physical world and the cyber world, enables the creation of a bunch of smart applications [5], such as smart city, smart industry, and smart health care system. Considering that most IoT devices are resource constrained and cannot handle the huge amount of data locally and efficiently, the cloud storage server is included in the IoT and forms a new paradigm, the cloud-IoT, where the cloud or a resource-adequated server provides useful services like storage and computing. ABE schemes with a single attribute authority do not adequately address the needs of the ubiquitous IoT devices properly. In [6], Chase first proposed a multiauthority ABE scheme. However, Chase's scheme still requires the trusted central authority (CA), which can decrypt any ciphertext that it wants to decrypt. Later, Chase et al. improve their scheme by removing the CA and achieve a truly decentralized ABE scheme [7].

In [8], Lewko et al. proposed a distributed ABE scheme, which not only realizes multiauthority attribute-based encryption (MAABE) but also proves the system security with dual system encryption methodology. Unfortunately, the application of ABE in IoT still faces an important challenge: IoT devices with limited resources cannot afford the huge number of bilinear pairing operations in ABE schemes. Therefore, Green et al. proposed an outsourced ABE scheme which ensures the data security while minimizing the computational burden of equipments [9].

In this paper, a multiauthority attribute-based encryption scheme with white-box traceability and verifiable outsourced decryption was proposed for cloud IoT. Compared with the existing ABE schemes, our scheme has the following contributions:

(i) As there is a great quantity of attributes used in the decryption key generation, each attribute authority controlls a set of disjoint attributes independently in our scheme. The central authority is only responsible for generating the public parameters, and the right to decide who can decrypt is hold by the data owners directly

(ii) Our scheme uses the linear secret sharing schemes (LSSS) to allow any monotone access structures. More importantly, to protect the privacy of IoT users, our scheme realizes fully hidden access policy

(iii) Considering the needs of resource-constrained IoT devices, our scheme outsources most decryption works to the cloud by the verifiable outsourcing technology

(iv) Our scheme adopts the Boneh-Boyen short signature algorithm to implement the user traceability mechanism. In other words, we use a white-box trace algorithm to tackle the private key leaking issue

1.1. Paper Organizaiton. Section 2 summarizes many related works, and Section 3 introduces all preliminaries of our scheme including some complexity assumptions. The system model and security models are presented in Section 4. In Section 5, we propose the concrete construction and a simple application of our scheme. Section 6 outlines the proof of indistinguishability, verifiability, fully hiding, and traceability of our scheme. We compare our scheme with some other schemes about the storage and computation costs in Section 7. Section 8 contains the conclusion.

2. Related Work

Many works have been proposed since Sahai et al. first proposed the attribute-based encryption [3]. ABE schemes can be classified into two categories generally: the key-policy attribute-based encryption (KP_ABE) and the ciphertext-policy attribute-based encryption (CP_ABE) [4, 16]. Because CP_ABE allows the data owner to decide the access policy, it has been treated as the most promising solution to solve the access control issue in the cloud storage. In ABE schemes, the key pair of data users is generated by attribute authorities (AAs). Thus, the security of ABE schemes is based on the trust of the attribute authorities. To tackle the huge amount of data users contained in IoT, multiauthority attribute-based encryption (MA_ABE) was proposed, which can manage the huge amount of attributes in a more efficient way [6, 17–19], where each attribute authority controls an unique set of attributes independently. To achieve both the data confidentiality and the data authentication in the body area network, Hu et al. proposed a fuzzy attribute-based signcryption scheme [20].

Another characteristic of IoT is that most devices are resource-limited [21–23]. As we all know that the decryption overhead of ABE schemes rises along with the attribute number involved in the access policy. Obviously the expensive pairing computations are unacceptable for most IoT devices. Therefore, some ABE schemes using the proxy reencryption concept have been proposed [24–26]. In [9], Green et al. proposed an outsourced ABE scheme, which outsources most decryption overheads to a trusted third-party server, but outsourced ABE schemes all rely on a semitrusted server to semidecrypt that leads to a serious problem: how to ensure the semidecrypted data is correct and not altered. In [27], Lai et al. proposed a verifiable outsourced ABE while this scheme requires heavy costs for decryption. Recently, Li et al. improved an ABE scheme to achieve not only verifiable outsourced decryption but also lightweight user decryption [10], but all outsourced schemes mentioned above rely on a central authority to manage and generate user decryption key. In [11], Belguith et al. proposed an outsourced multi-authority attribute-based encryption scheme. In [28], Deng et al. proposed an efficient outsourced attribute-based signcryption scheme which also solves the user revocation problem.

In the cloud-assisted IoT environment, data owners store private data in the shared cloud. In most ABE schemes, the access policy is uploaded to the cloud server in plaintext along with the encrypted data. This may reveal private information of the encryptor and the decrypor. In [29], Nishide et al. proposed an ABE scheme with partially hided access policy, but this scheme has poor expressiveness.

When it comes to application in the real word, a common issue of ABE schemes needs to be considered: the leakage of

decryption keys. In other words, how to trace/recover the global identity of the guilty user who leaks its secret key to a malicious or illegal user. There are two tracing approaches, white-box traceability and black-box traceability, that can be used to solve this issue. In [30], Hinek et al. used the Boneh-Boyen signature [31] to achieve the white-box traceability. Liu et al. proposed a white-box traceable ABE [32] and a black-box traceable ABE with highly expression [33]. In [12], Liu et al. proposed a traceable and revocable ABE scheme which is more practical for real application. In [34], Yu et al. proposed a traceable ABE scheme with white-box traceability to manage data stored in the cloud storage. In [13], an efficient large-universe MA_CP_ABE with white-box traceability was proposed. While in [35], Qiao et al. proposed a traceable ABE scheme with black-box traceability for fog computing.

All traceable ABE schemes mentioned above have a shared issue: their decryption computation burden are intolerable for IoT devices. In [14], an ABE scheme with outsourced decryption designed for electronic health systems was proposed by Li et al. However, Li's scheme did not consider the privacy of access policies which might contain sensitive personal information of users. We compare our scheme with some existed ABE schemes in Table 1. In a word, our ABE scheme achieves selective replayable CCA security and provides multiple practical functions, such as fully hidden policy, outsourced decryption, and traceability.

3. Preliminaries

In this section, we provide all mathematical preliminaries needed for our scheme.

3.1. Bilinear Maps. Let $\mathbb{G}$ and $\mathbb{G}_T$ be two multiplicative cyclic groups of prime order p. Let g be a generator of $\mathbb{G}$ and e be a bilinearmap, $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, with the following three properties [15]:

(1) Bilinearity: for all $u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p$, we have $e(u^a, v^b) = e(u, v)^{ab}$, where $\mathbb{Z}_p$ is the integers modulo p

(2) Nondegeneracy: $e(g, g) \neq 1$, where 1 is the unit of $\mathbb{G}_T$

(3) Computability: there is a polynomial time algorithm to efficiently compute $e(u, v)$ for any $u, v \in \mathbb{G}$

We say $\mathbb{G}$ is a bilineargroup if the group operation in $\mathbb{G}$, and the bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is both efficiently computable. Notice that the map e is symmetric since $e(g^a, g^b) = e(g, g)^{ab} = e(g^b, g^a)$.

3.2. Access Structure

Definition 1. (access structure). Let $P = \{P_1, \cdots P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1, \cdots P_n\}}$ is monotone if $\forall B, C$: *if* $B \in \mathbb{A}$ and $B \subseteq C$, then $C \in \mathbb{A}$. An access structure is a collection $\mathbb{A}$ of nonempty subsets of $\{P_1, \cdots P_n\}$, such as $\mathbb{A} \subseteq 2^{\{P_1, \cdots P_n\}} \setminus \varnothing$. The sets in $\mathbb{A}$ are called authorized sets, and the sets not in $\mathbb{A}$ are called unauthorized sets [9].

3.3. Linear Secret Sharing Schemes (LSSS)

Definition 2. (linear secret sharing schemes (LSSS)). A secret-sharing scheme $\prod$ over a set of parties $\mathbb{P}$ is called linear over $\mathbb{Z}_p$ if

(1) The shares of a secret $s \in \mathbb{Z}_p$ for each party form a vector over $\mathbb{Z}_p$

(2) There exists a matrix $\mathbb{M}$ with l rows and n columns called the share-generating matrix for $\prod$ and a function ρ which maps each row of the matrix to an associated party. That is, for $i = 1, .., l$, the value $\rho(i)$ is the party associate with the row i. When we consider the column vector $v = (s, r_2, \cdots, r_n)$ where $r_2, \cdots, r_n \in \mathbb{Z}_p$ is randomly chosen, then $\mathbb{M}v$ is the vector of l shares of the secret s according to $\prod$. The share $(\mathbb{M}v)_i$ belongs to the party $\rho(i)$

According to [9], every linear secret-sharing scheme based on the above definition also enjoys the linear reconsruction property defined as follows: Let $\prod$ be an LSSS for the access structure $\mathbb{A}$. Let $S \in \mathbb{A}$ be any authorized set, and let $I \subset \{1, 2, \cdots, l\}$ be defined as $I = \{i : \rho(i) \in S\}$. Then, there exist constants $\{\omega_i \in \mathbb{Z}_p\}_{i \in I}$ such that if $\{\lambda_i\}$ are valid shares of any secret s according to $\prod$, then $\sum_{i \in I} \omega_i \lambda_i = s$. It is shown in [9] that these constants $\{\omega_i\}$ can be found in polynomial time in the size of the share-generating matrix $\mathbb{M}$.

3.4. One-Way Anonymous Key Agreement. One-way anonymous key agreement [15] scheme can be used to guarantee anonymity of the access structure. This scheme only ensures the anonymity of one participant. Assume that there are two participants Alice (ID_A) and Bob (ID_B) in this scheme. And the master secret of the key generation center (KGC) is s. When Alice wants to keep anonymity, the process is listed as follows:

(1) Alice calculates $Q_B = H(ID_B)$. A random number $r_a \in \mathbb{Z}_p^*$ is choosed to generate the pseudonym $P_A = Q_A^{r_a}$ and computes the session key $K_{A,B} = e(d_A, Q_B)^{r_a} = e(Q_A, Q_B)^{s \cdot r_a}$. Finally, she sends her pseudonyms P_A to Bob

(2) Bob uses his secret key d_B to calculate the session key $K_{A,B} = e(P_A, d_B) = e(Q_A, Q_B)^{s \cdot r_a}$, where $d_i = H(ID_i)^s \in \mathbb{G}$ is his private key for $i \in \{A, B\}$, and $H : \{0, 1\}^* \to \mathbb{G}$ is a strong collision-resistant hash function

3.5. Complexity Assumptions

Definition 3. Strong Diffie Hellman problem (q-SDH). Let $\mathbb{G}$ be a multiplicative cyclic group of order p with a generator g. Given a random $x \in \mathbb{Z}_p^*$ and a $q + 1$ tuple $(g, g^x, g^{x^2}, \cdots, g^{x^q})$,

Table 1: Function comparison.

Scheme	[10]	[11]	[12]	[13]	[14]	[15]	Our scheme
Multiauthority	No	Yes	No	Yes	No	Yes	Yes
Access policy	AND gates	LSSS	LSSS	LSSS	LSSS	LSSS	LSSS
Fully hidden policy	No	No	No	No	No	Yes	Yes
Outdecryption	Yes	Yes	No	No	Yes	No	Yes
Security	RCCA	Selective RCPA	Selective CPA	Static security	Selective CPA	Selective CPA	Selective RCCA
Traceability	No	No	Yes	Yes	Yes	No	Yes

the problem of computing a pair $(c, g^{1/x+c})$, where $c \in \mathbb{Z}_p^*$, is called the q-strong Diffie Hellman problem [13].

Definition 4. Computational Diffie Hellman problem (CDH). Let $\mathbb{G}$ be a multiplicative cyclic group of order p with a generator g. Given two group elements $g^a, g^b \in \mathbb{G}$ where $a, b \in \mathbb{Z}_p$ are two random integers. The problem of calculating g^{ab} from g^a and g^b is called Computational Diffie Hellman problem [11].

Definition 5. Decisional Bilinear Diffie Hellman problem (DBDH). Let $\mathbb{G}$ be a multiplicative cyclic group of order p with a generator g. Given three group element g^a, g^b, and $g^c \in \mathbb{G}$ where $a, b,$ and $c \in \mathbb{Z}_p^*$ are three random integers. The problem of distinguishing tuples of the form $(g^a, g^b, g^c, e(g,g)^{abc})$ *and* $(g^a, g^b, g^c, e(g,g)^z)$ for some random integer z is called the Decisional Bilinear Diffie Hellman problem [11].

4. System Definition

4.1. System Model. The system model of our scheme is illustrated in Figure 1, and the associated five entities are described as follows:

(1) Central Trusted Authority (CTA): the CTA is only used to generate the public parameter, and it cannot decrypted any data

(2) Attribute authorities (AAs): each AA controls a set of attributes. Multiple attribute authorities work together to generate the user's decryption key. Besides, attribute authorities can use a trace algorithm to recover the global identity of the guilty user who leaks its private decryption key

(3) Cloud storage service provider (CS): the CS is responsible to store the encrypted data. Moreover, CS performs the outsourcing decryption for users

(4) Data owner (DO): the owner of the data which is responsible to encrypt and upload the data to CS

(5) Data user (DU): the party who wants to access data

Table 2 summarizes notations used in our scheme. Assume that there are n authorities in our scheme and each attribute is associated with an unique AA, such that $S_{A,AA_i} \cap S_{A,AA_j} = \varnothing$ for $\forall i, j$ and $\bigcup S_{A,AA_j} = S_A$.

4.2. System Procedure. Our MAABE scheme with outsourced decryption and hidden policy contains the following five phases:

(1) *System initialization* : this phase includes two algorithms. Firstly, the CTA runs the $setup(\lambda) \rightarrow \mathrm{PP}$ algorithm to generate the global parameters PP, where λ is the security parameter. Then, each AA runs the $\mathrm{Setup}_{\mathrm{auth}}(\mathrm{PP}) \rightarrow (\mathrm{sk}_{\mathrm{AAj}}, \mathrm{pk}_{\mathrm{AAj}})$ algorithm to generate their own key pairs, which is consisted with a private key and a public key

(2) *Encryption* : the DO runs the $\mathrm{Encrypt}(\mathrm{PP}, \{\mathrm{pk}_{\mathrm{AAj}}\}, \mathrm{MSG}, (\mathbb{M}, \rho)) \rightarrow \mathrm{CT}$ algorithm to encrypt the message MSG, and then it uploads the ciphertext to the cloud server

(3) *Keygeneration* : this phase contains two algorithms. Firstly, each related AA runs the $\mathrm{Keygen}(\mathrm{PP}, \{sk_{\mathrm{AAj}}, \mathrm{pk}_{\mathrm{AAj}}\}, \mathrm{GID}, \mathrm{S}_{\mathrm{GID,j}}) \rightarrow \mathrm{sk}_{\mathrm{GID,j}}$ algorithm independently to generate the decryption key for the DU with identity GID. Then, all results are sent to the user

To outsource the decryption work to the cloud, the user runs the $\mathrm{Keygen}_{\mathrm{out}}(\mathrm{PP}, \mathrm{sk}_{\mathrm{GID}}, (\mathbb{M}, \rho), \mathrm{CT}) \rightarrow \mathrm{ok}_{\mathrm{GID}}$ algorithm to generate its outsourced decryption key.

(4) *Decryption* : this phase is divided into two steps. Firstly, the CS runs the $\mathrm{Decrypt}_{\mathrm{out}}(\mathrm{PP}, \mathrm{opk}_{\mathrm{GID}}, (\mathbb{M}, \rho), \mathrm{CT}) \rightarrow \mathrm{CT}'$ algorithm to partially decrypt the ciphertext. The second step is performed by the user, who runs the $\mathrm{Decrypt}(CT', osk_{\mathrm{GID}}) \rightarrow \mathrm{MSG}$ algorithm to get the plaintext

(5) *Trace* : to begin with, each AA_j verifies the format of the decryption key that needed to be traced, and then it runs the $\mathrm{Trace}(PP, sk_{\mathrm{GID}}, \{pk_{AAj}\}) \rightarrow \mathrm{GID}$ algorithm to output the global identity (GID) of the guilty user

4.3. Security Models. We define four security models of our MAABE scheme in this section.

(1) *Confidentiality:* the confidentiality of data is the basic security requirement of a scheme, which is used to resist malicious adversaries to gain extral information from

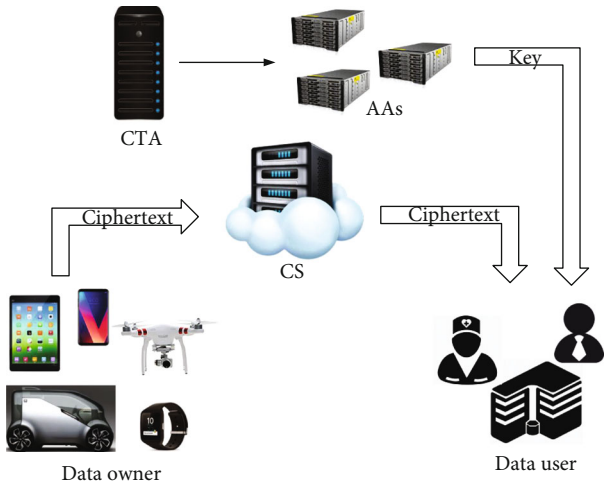

Figure 1: System model and procedure.

Table 2: Notations.

Notation	Meaning
S_{AA}	The universe set of attribute authorities
S'_{AA}	The set of corrupted attribute authorities
$S_{AA,GID}$	The set of attribute authorities related to the user GID
S_A	The universe set of attributes
AA_j	An attribute authority
S_{A,AA_j}	The attribute set controlled by AA_j
GID	The global identity of a user
S_{GID}	The attribute set related with user GID
P	The public parameter
sk_{AA_j}	Secret key related to AA_j
pk_{AA_j}	Public key related to AA_j
MSG	Message
$(\mathbb{M}, \rho)$	The LSSS access matrix and its row label function
ϕ	Access control structure
S_ϕ	The attribute set related with ϕ
CT	The ciphertext
sk_{GID}	The secret key of user GID
ok_{GID}	The outsourced decryption key ([opk, osk])

the ciphertext. Our scheme adopts the replayable chosen-ciphertext security (RCCA) defined in [36] by Canetti et al. as this type of security is sufficient enough and not to be too strict. Two restrictions are followed in this experiment: all decryption key queries cannot satisfy the challenge access struction fixed in the initialization phase by the adversary. And the attribute authorities can only be corrupted statically be the adversary

The selective secure against chosen ciphertext attack of our scheme is achieved if no probabilistic polynomial time (PPT) adversary can win the Exp^{Conf} security experiment described in Figure 2 between an adversary $\mathbb{A}$ and a challenger $\mathbb{C}$ with nonnegligible advantage.

(2) *Verifiability*: our scheme is verifiable if there is no PPT adversary that can win the $\text{Exp}^{\text{Verif}}$ security experiment described in Figure 3 between an adversary $\mathbb{A}$ and a challenger $\mathbb{C}$ with nonnegligible advantage.

(3) *Fully hidden:* in our scheme, the CS knows nothing about the access policy, and the user only knows if his/her attributes satisfy the access policy. Our scheme is an outsourced ABE with fully hidden policy if there is no PPT adversary that can win the Exp^{Hide} security experiment described in Figure 4 between an adversary $\mathbb{A}$ and a challenger $\mathbb{C}$ with nonnegligible advantage. The goal of the adversary is to recover the correct access policy without the required decryption key.

(4) *Traceability:* our scheme is a traceable ABE if there is no PPT adversary can win the $\text{Exp}^{\text{Trace}}$ security experiment described in Figure 5 between an adversary $\mathbb{A}$ and a challenger $\mathbb{C}$ with nonnegligible advantage.

Definition 6. An outsourced ABE scheme is RCCA-secure against static corruption of the attribute authorities if $Adv_{\mathbb{A}}[\text{Exp}^{\text{Conf}}(1^\xi)]$ is negligible for all PPT adversaries.

Definition 7. An outsourced ABE scheme is verifiable if $Adv_{\mathbb{A}}[\text{Exp}^{\text{Verif}}(1^\xi)]$ is negligible for all PPT adversaries.

Definition 8. An outsourced ABE scheme achieves policy private if $Adv_{\mathbb{A}}[\text{Exp}^{\text{Priv}}(1^\xi)]$ is negligible for all PPT adversaries.

Definition 9. An outsourced ABE scheme is traceable if $Adv_{\mathbb{A}}[\text{Exp}^{\text{Trace}}(1^\xi)]$ is negligible for all PPT adversaries.

5. Construction and Application

The concrete construction of our MAABE scheme is presented in this section. Firstly, the CTA and all AA perform initialization and generate the PP and the public keys of AAs. Then, the DO can encrypt its data with an access structure. Before accessing the data, the DU needs to request its decryption key to the AAs. Next, the DU can access the data and decrypt successfully with the help of the cloud predecrypting for the DU first. Finally, the trace algorithm is used to reform the global identity of a guilty data user by the AAs. This section also contains a simple application in the end.

5.1. Concrete Construction

5.1.1. Phase I: System Initialization

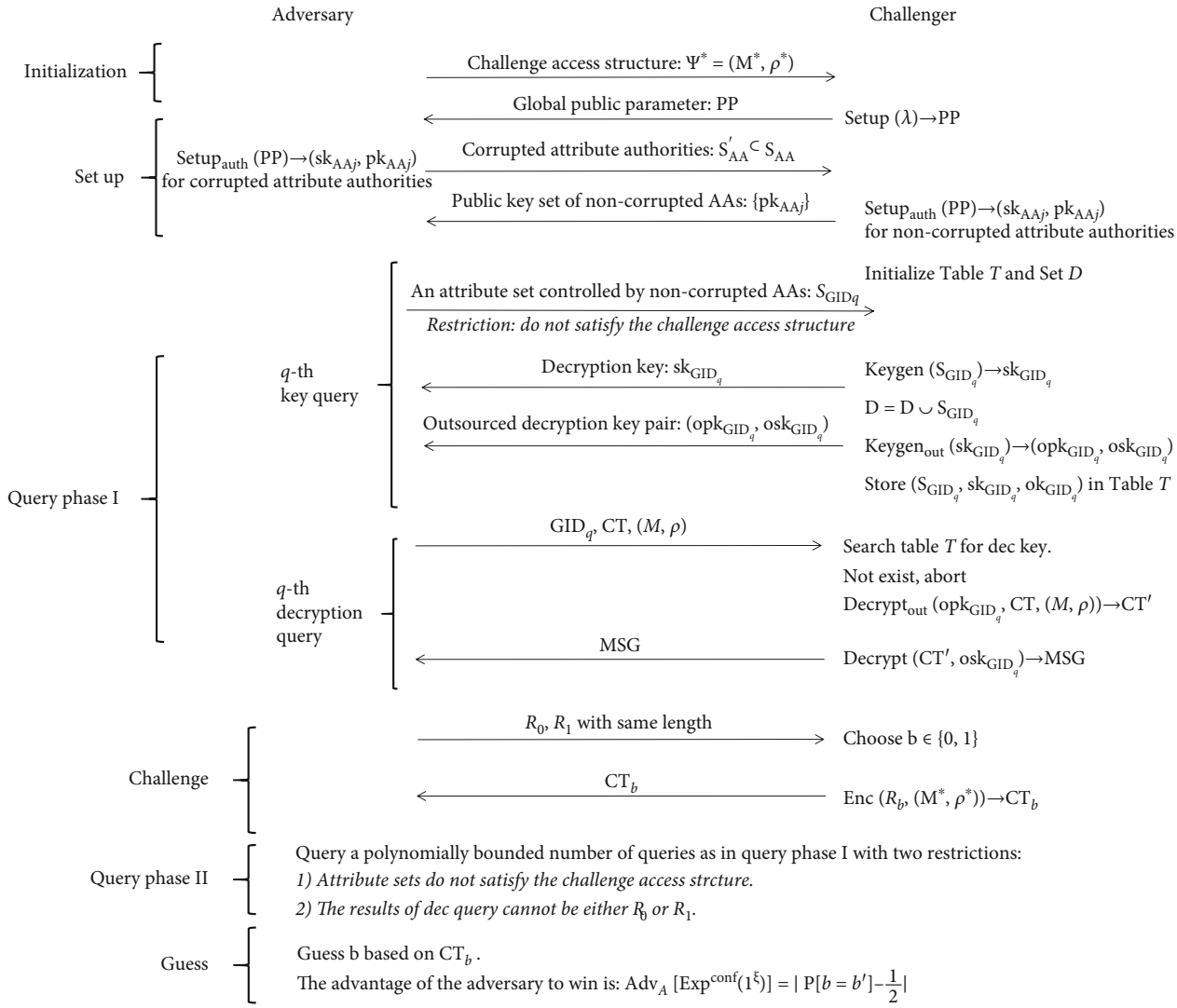

Figure 2: Confidentiality security model.

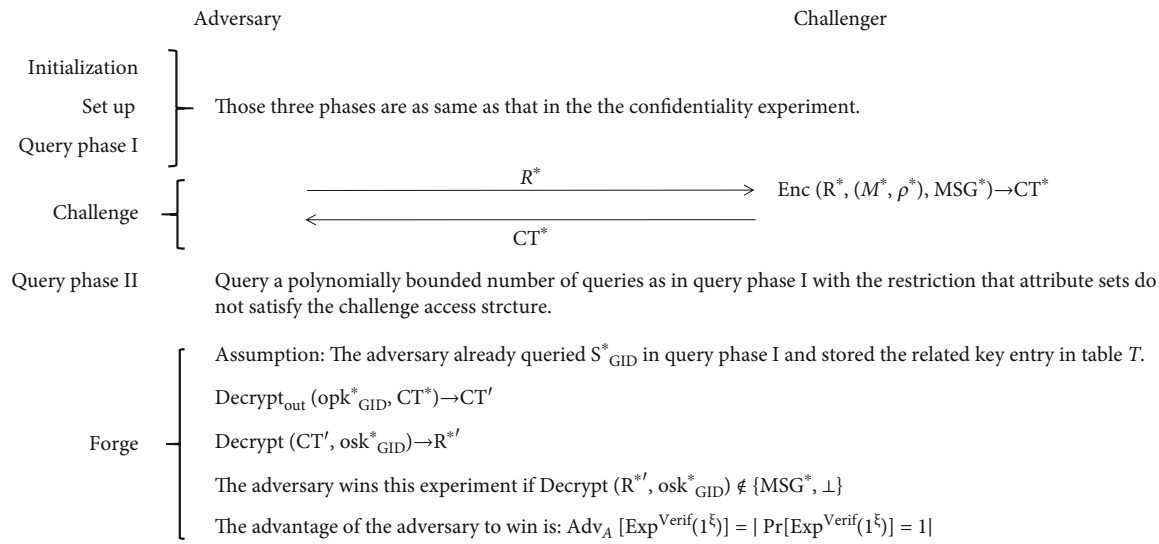

Figure 3: Verifiability security model.

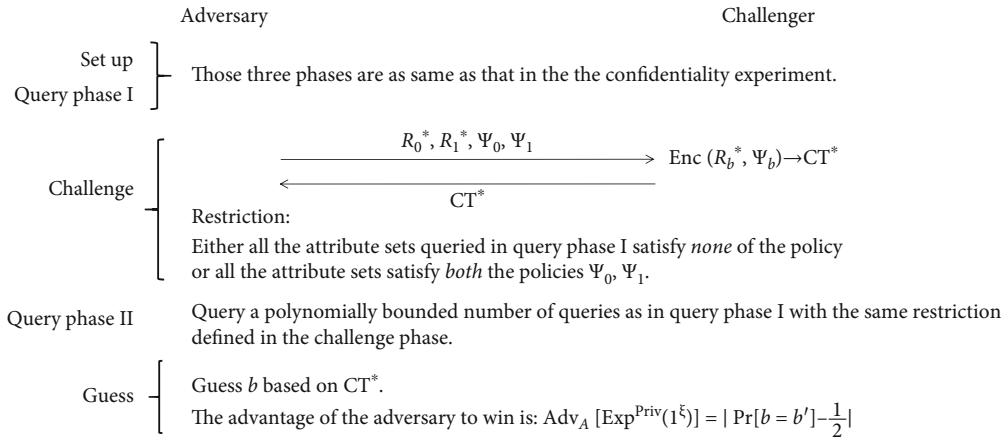

Figure 4: Fully hidden security model.

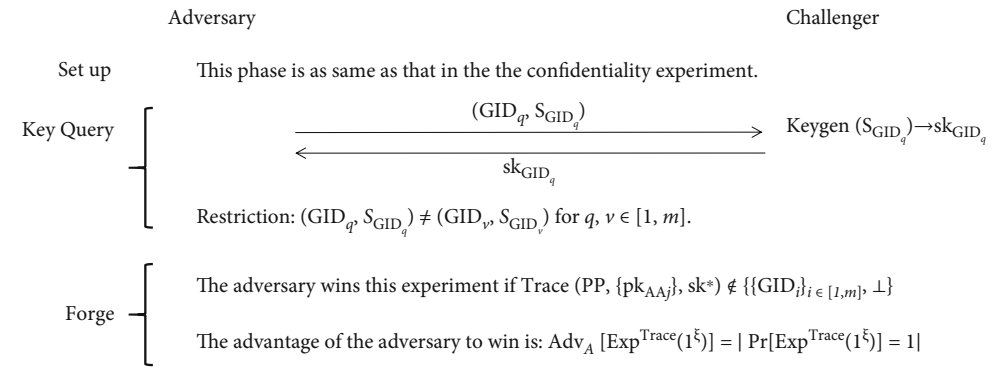

Figure 5: Traceability security model.

(1) *System set – up* : this step is performed by the CTA

It defines two multiplicative group $\mathbb{G}, \mathbb{G}_T$ of prime order p, and g is a generator of $\mathbb{G}$.

It defines a symmetric bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$.

It defines three collusion resistant hash functions as follows: $H : \{0.1\}^* \rightarrow \mathbb{G}, H_1 : \{0,1\}^* \rightarrow \mathbb{Z}_p^*, H_2 : \{0,1\}^* \rightarrow \{0,1\}^k$ where k is the length of the symmetric key.

It defines a CPA-secure symmetric encryption scheme $(\mathrm{Enc_{sym}}, \mathrm{Dec_{sym}})$.

It outputs the global pubic parameter PP:

$$PP = \left\{\mathbb{G}, \mathbb{G}_T, p, e, g, H, H_1, H_2, \left(\mathrm{Enc_{sym}}, \mathrm{Dec_{sym}}\right)\right\} \quad (1)$$

(2) *Authority set – up* : each attribute authority performs this step to get their key pair. We take the A A_j as an example

It chooses two random numbers $\alpha_i, \beta_i \in \mathbb{Z}_p^*$ for each attribute $i \in S_{A,AA_j}$.

It chooses three random numbers $h_j, a_j, b_j \in \mathbb{Z}_p^*$.

It generates its pair of private key sk_{AAj} and public key pk_{AAj} as follows:

$$\begin{cases} sk_{AAj} = \left(\{\alpha_i, \beta_i\}_{i \in S_{AAj}}, h_j, a_j, b_j\right), \\ pk_{AAj} = \left(\left\{g^{\alpha_i}, g^{\beta_i}\right\}_{i \in S_{AAj}}, g^{h_j}, g^{a_j}, g^{b_j}\right). \end{cases} \quad (2)$$

5.1.2. Phase II: Encryption. We assume that the DO encrypts a message MSG with an self-defined access structure Ψ, and S_Ψ is the attribute set which contains all attributes in the access structure Ψ. This phase contains three steps defined below:

(1) *Fully Hide* the *access policy*

It chooses a random number $a \in \mathbb{Z}_p^*$ and then computes $q_i = e((g^{h_j})^a, H(i))$ where $i \in S_\Psi$.

It replaces each attribute in S_Ψ with the corresponding q_i.

It converts the access policy to a LSSS access matrix $(\mathbb{M}_{l \times n}, \rho)$.

(2) *Encrypt* the *key seed*

It chooses a random element $R \in \mathbb{G}_T$ (the key seed) to calculate $s = H_1(R, MSG)$ and the symmetric key $Ksym = H_2(R)$.

It selects a $p_i \in \mathbb{Z}_p$ for each row $\mathbb{M}_i$ of $\mathbb{M}$ and two random vectors $\vec{v} = [s, v_1, \cdots, v_n] \in \mathbb{Z}_p^n$, $\vec{w} = [0, w_1, \cdots, w_n] \in \mathbb{Z}_p^n$.

It computers $\lambda_i = \mathbb{M}_i \times \vec{v}$ and $w_i = \mathbb{M}_i \times \vec{w}$.

It outputs the tuple $CT_{ABE} = (h, (\mathbb{M}_{l\times n}, \rho), C_0, \{C_{1,i}, C_{2,i}, C_{3,i}, C_{4,i}, C_{5,i}\}_{i\in[1,l]})$ where i presents a matrix row corresponding to an attribute.

Details of the ciphertext are presented as follows:

$$\mathrm{CT} = \begin{cases} h = g^a, \\ C_0 = \mathrm{Re}\ (g, g)^s, \\ C_{1,i} = g^{\lambda_i} g^{\alpha_{\rho(i)} p_i}, \\ C_{2,i} = g^{p_i}, \\ C_{3,i} = g^{w_i} g^{\beta_{\rho(i)} p_i}, \\ C_{4,i} = g^{a_j p_i}, \\ C_{5,i} = g^{b_j p_i} \end{cases} . \tag{3}$$

(3) *Encrypt the message*

Uses K_{sym} to encrypt the message MSG by the symmetric encryption algorithm Enc_{sym} and denote the result as $CT_{sym} = Enc_{sym}(K_{sym}, MSG)$.

It uploads $CT = \{CT_{ABE}, CT_{sym}\}$ to the CS.

5.1.3. Phase III: Key Generation

(1) Decryption key

Each user owns an unique global identity $GID \in \mathbb{Z}_p^*$ and an attribute set S_{GID} where each attribute is associated with a designed attribute authority. Let $S_{AA,GID}$ be the set of related attribute authorities. According to $S_{AA,GID}$, we divide S_{GID} into $\{S_{GID,j}\}_{j\in S_{AA,GID}}$. When the user queries its decryption key, each related AA runs the key generation algorithm. We take the AA_j as an instance.

It chooses a random number $r \in \mathbb{Z}_p^* \setminus \{-a_j + GID/b_j\}$ for each $i \in S_{GID,j}$.

It computes and returns the decryption key $sk_{GID,j} = \{K_{1,i}, K_{2,i}, K_{3,i}\}_{i\in S_{GID}}$:

$$\begin{cases} K_{1,i} = g^{\alpha_i/a_j + GID + b_j r} H(GID)^{\beta_i/a_j + GID + b_j r}, \\ K_{2,i} = H(i)^{h_j}, \\ K_{3,i} = r. \end{cases} \tag{4}$$

The decryption key of the user GID is noted as

$$sk_{GID} = \left(\{sk_{GID,j}\}_{j\in S_{AA,GID}}, GID\right) = \left(\{K_{1,i}, K_{2,i}, K_{3,i}\}_{i\in S_{GID}}, GID\right) \tag{5}$$

(2) Outsourced decryption key: the data user runs this algorithm

(a) Reconstructs the access policy

It computes $q_i' = e(h, H(i)^{h_j}) = e(g^a, H(i)^{h_j}), \forall i \in S_{GID}$.

It uses q_i' to replace the attribute i to get the attribute set S_{GID}'.

It gains the access structure $(\mathbb{M}_{l\times n}, \rho)$ from CT.

It identifies the set of attributes $L' = \{i : (\rho(i) \cap S_{GID}')_{i\in[l]}\}$ required for the decryption.

(b) Generates the outdec key

Chooses a random number $z \in \mathbb{Z}_p^*$ to compute the outsourced decryption key $\{ok_{GID}\} = (\{opk_{GID}\}, osk_{GID})$ as

$$\begin{cases} opk_{GID} = \left(GID, \left\{K_{1,i}^{1/z}, K_{3,i}\right\}_{i\in L'}, g^{1/z}, H(GID)^{1/z}\right), \\ osk_{GID} = z. \end{cases} \tag{6}$$

5.1.4. Phase IV: Decryption

(1) *Outsourced decryption* : the CS performs outsourced decryption for the user

It computes the following equation for each matrix row corresponding to an attribute i:

$$Q = \frac{e\left(g^{1/z}, C_{1,i}\right) e\left(H(GID)^{1/z}, C_{3,i}\right)}{e\left(K_{1,i}^{1/z}, C_{2,i}^{GID} C_{4,i} C_{5,i}^{K_{3,i}}\right)} = \left(e(g, g)^{\lambda_i} e(H(GID), g)^{w_i}\right)^{1/z}. \tag{7}$$

It chooses a set of constants $\{c_i\}_{i\in[1,l]} \in \mathbb{Z}_p$ such that $\sum_i c_i \mathbb{M}_i = [1, 0, \cdots, 0]$.

It Computes

$$\prod_{i=1}^{l} Q^{c_i} = \begin{pmatrix} e(g, g)^{\sum_{i=1}^{l} \lambda_i c_i} & e(g, H(GID))^{\sum_{i=1}^{l} w_i c_i} \end{pmatrix}^{1/z}, \tag{8}$$

where l is the row number of the access matrix.

It returns $CT' = \prod_{i=1}^{l} Q^{c_i} = e(g, g)^{s/z}$ to the user.

(2) *Uer decryption* : this phase contains the following two steps

(a) Recovers the message R based on the partially decrypted ciphertext CT' by computing the following equation

$$\mathrm{R} = \frac{C_0}{\left(CT'\right)^{osk}} = \frac{C_0}{\left(e(g,g)^{s/z}\right)^{z}} = \frac{C_0}{e(g,g)^{\mathrm{s}}}. \tag{9}$$

(note that this equation costs one exponentiation only and no pairing performance.)

(b) Computes $\mathrm{K_{sym}} = \mathrm{H_2(R)}$, $\mathrm{MSG} = \mathrm{Dec_{sym}(K_{sym}, C\ T_{sym})}$, and $\mathrm{s} = \mathrm{H_1(R, MSG)}$

Judge if $\mathrm{CT'} = \mathrm{e(g, g)^{s/z}}$. If no, outputs $\perp$. Else, the user gains the right MSG.

Correctness of Equation (7):

Proof. First for each attribute $\mathrm{i} \in L'$, the CS uses $(\{opk_{GID}\}$ to compute:

$$\begin{aligned}\mathrm{Q} &= \frac{e\left(g^{1/z}, g^{\lambda_i} g^{\alpha_{\rho(i)} p_i}\right) e\left(H(\mathrm{GID})^{1/z}, g^{\beta_{\rho(i)} p_i} g^{w_i}\right)}{e\left(g^{\alpha_{\rho(i)}/z\left(a_j + GID + b_j r\right)} H(\mathrm{GID})^{\beta_{\rho(i)}/z\left(a_j + GID + b_j r\right)}, \left(g^{p_i}\right)^{GID} g^{a_j p_i} g^{b_j p_i r}\right)} \\ &= \frac{e(g,g)^{\lambda_i/z} e(g,g)^{\alpha_{\rho(i)} p_i/z} e(g, H(\mathrm{GID}))^{\beta_{\rho(i)} p_i/z} e(g, H(\mathrm{GID}))^{w_i/z}}{e(g,g)^{\alpha_{\rho(i)} p_i/z} e(g, H(\mathrm{GID}))^{\beta_{\rho(i)} pi/z}} \\ &= e(g,g)^{\lambda_i/z} e(g, H(\mathrm{GID}))^{w_i/z}.\end{aligned} \tag{10}$$

Then, it chooses a set of constants $\{c_i\}_{i \in [1,l]} \in \mathbb{Z}_p$ such that $\sum_{\mathrm{i}} c_i \mathbb{M}_i = [1, 0, \cdots, 0]$. Because $\lambda_i = \mathbb{M}_i \vec{v}$ and $\mathrm{w_i} = \mathbb{M}_i \vec{w}$, so

$$\begin{aligned}\sum_{\mathrm{i}=1}^{\mathrm{l}} \lambda_i c_i &= \sum_{\mathrm{i}=1}^{\mathrm{l}} \mathbb{M}_i \vec{v} c_i = \vec{v}[1, 0, \cdots, 0] = s, \\ \sum_{\mathrm{i}=1}^{\mathrm{l}} w_i c_i &= \sum_{\mathrm{i}=1}^{\mathrm{l}} \mathbb{M}_i \vec{w} c_i = \vec{w}[1, 0, \cdots, 0] = 0.\end{aligned} \tag{11}$$

Hence, we can get

$$\begin{aligned}CT' &= \prod_{i=1}^{l} Q^{c_i} = \prod_{i=1}^{l} e(g,g)^{c_i \lambda_{\rho(i)}/z} e(g, H(\mathrm{GID}))^{c_i w_i/z} \\ &= e(g,g)^{\sum_{i=1}^{l} c_i \lambda_{\rho(i)}/z} e(g, H(\mathrm{GID}))^{\sum_{i=1}^{l} c_i w_i/z} \\ &= e(g,g)^{s/z} e(g, H(\mathrm{GID}))^{0} = e(g,g)^{s/z}.\end{aligned} \tag{12}$$

Then, based on CT', the user recovers

$$R = \frac{C_0}{\left(CT'\right)^{osk}} = \frac{\mathrm{Re}\ (g,g)^{s}}{\left(e(g,g)^{s/z}\right)^{z}} = \frac{\mathrm{Re}\ (g,g)^{s}}{e(g,g)^{s}}. \tag{13}$$

5.1.5. Phase V: Trace. The $\mathrm{Trace(PP}, sk_{\mathrm{GID}}, \{pk_{\mathrm{AA}j}\})$ algorithm is performed by all attribute authorities. The input is the private key $\mathrm{sk_{GID}} = (\{\mathrm{sk_{GID,j}}\}_{\mathrm{j} \in S_{\mathrm{AA}}'}, \mathrm{GID}) = (\{\mathrm{K_{1,i}}, \mathrm{K_{2,i}}, \mathrm{K_{3,i}}\}_{\mathrm{i} \in S_{\mathrm{GID}}}, \mathrm{GID})$ of a user.

Firstly, the AA checks the form of the key. If the key does not satisfies the form, this algorithm aborts.

Then, the AA searches its database to find if $\exists i \in S_{GID}$, s.t.

$$\begin{aligned}&\mathrm{K_{1,}}_i, K_{2,i} \in \mathbb{G}, K_{3,i}, GID \in \mathbb{Z}_p^*, \\ &\mathrm{e}\left(K_{1,i}, g^{a_j} g^{\left(b_j\right)^{K_{3,}}{}_i} g^{\mathrm{GID}}\right) = e(g,g)^{\alpha_i} e\left(H(\mathrm{GID}), g^{\beta_i}\right).\end{aligned} \tag{14}$$

If yes, the global identity GID of the guilty user will be output.

5.2. Application in the EHR System. In this section, we describe a simple application of our scheme based on the electronic health record (EHR) system. The basic procedures are presented in Figure 6, and the details are described as follows:

(1) The central trusted authority (the government) performs the system set-up algorithm to generate and publish the global parameters PP

(2) A set of management companies act as attribute authorities, and each attribute authority needs to set up first. Then, they publish their public keys while keeping private keys secret

(3) A hospital encrypts a patient's medical records Rd based on a user-defined access structure $\mathbb{M}$ and sends the ciphertext CT along with the fully hidden access structure n to the cloud storing server to store

$$\mathrm{Hospital} \xrightarrow{\mathrm{Encrypt(Rd,\mathbb{M}) \to CT}} \mathrm{Cloud}. \tag{15}$$

(4) Before a data user (a doctor) requests the wanted records from the cloud server, he/she needs to get the decryption key sk_{GID} from the attribute authorities first

(5) To outsource the decryption work to the cloud server, the doctor generates the outsourced decryption key opk_{GID} based on sk_{GID}. Then, he/she sends opk_{GID} to the cloud server

(6) The cloud server will partially decrypt for the doctor as long as his/her attribute set satisfies the encryption

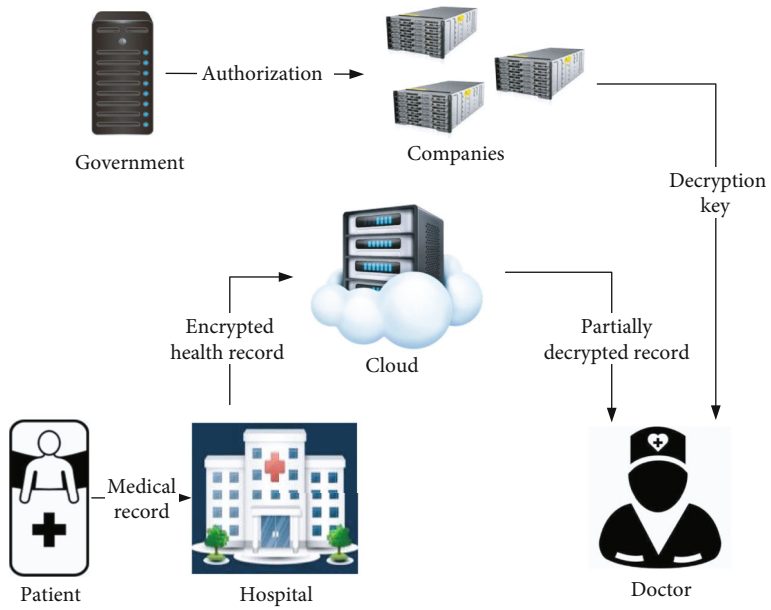

Figure 6: Application in an EHR system.

access structure. Then, the cloud sends the partially decrypted ciphertext CT' back to the doctor

$$\text{Cloud} \xrightarrow{\text{Decrypt}_{\text{out}}(\text{opk}_{\text{GID}},\text{CT})\rightarrow\text{CT}} {}'\text{Doctor}. \tag{16}$$

(7) Finally, the doctor can fully decrypt and get the medical records. Note that doctors only require one exponentiation in $\mathbb{G}_T$ to fully decrypt, and one hash operation to verify whether the ciphertext was tampered

$$\text{Decrypt}\left(\text{CT}'\right) \rightarrow \text{Rd}. \tag{17}$$

(8) If a malicious user decrypt illegally with a valid decryption key, the attribute authorities can perform the trace algorithm to recover the identity of the guilty user who leaks his/her decryption key to a illegal user

6. Security Analysis

6.1. Indistinguishability

Theorem 1. *If Lewko et al.'s scheme [8] is CPA=secure, our multiauthority attribute-based encryption scheme is selectively replayable CCA-secure according to Definition 6 such that* $Adv_A[Exp^{Conf}] < Adv_A[Exp^{Lewko}]$.

Proof. We define a PPT adversary $\mathbb{A}$ running the experiment defined in Section 4.3(1) with an entity $\mathbb{B}$. $\mathbb{B}$ running Lewko et al.'s CPA-secure [8] experiment with a challenger $\mathbb{C}$. The proof described below is going to show that the advantage of $\mathbb{A}$ to win the experiment Exp^{Conf} is smaller than the advantage of $\mathbb{B}$ to win Lewko et al.'s CPA-secure experiment $\text{Exp}^{\text{Lewko}}$. The detailed interactions are described as follows:

(1) Initialization: the adversary $\mathbb{A}$ submits a challenge access policy $\Psi^* = (\mathbb{M}^*, \rho^*)$ to the challenge $\mathbb{C}$ through $\mathbb{B}$

(2) Set-Up: $\mathbb{C}$ runs the Setup(λ) algorithm to generate the global parameter PP

It chooses two multiplicative cyclic groups $\mathbb{G}, \mathbb{G}_T$ of prime order p with a generator g of $\mathbb{G}$.

It chooses a bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$.

It chooses three collusion-resistant hash functions $H^* : \{0,1\}^* \rightarrow \mathbb{G}$, $H_1^* : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$, $H_2^* : \{0,1\}^* \rightarrow \{0,1\}^k$.

It chooses a cpa-secure symmetric encryption scheme (Enc_{sym}, Dec_{sym}).

It sends the global parameter $\text{PP} = \{\mathbb{G}, \mathbb{G}_T, p, e, g, H^*, H_1^*, H_2^*, (\text{Enc}_{\text{sym}}, \text{Dec}_{\text{sym}})\}$ to $\mathbb{A}$ through $\mathbb{B}$.

It runs the $\text{Setup}_{\text{auth}}$ algorithm to generate the key pairs of the noncorrupted authorities:

It chooses two random numbers α_i and $\beta_i \in \mathbb{Z}_p^*$ for each attribute $i \in S_{A,AA_j}$.

It chooses three random numbers h_j, a_j, and $b_j \in \mathbb{Z}_p^*$ to compute the public key $\text{pk}_{AA_j} = (\{g^{\alpha_i}, g^{\beta_i}\}_{i \in S_{A,AA_j}}, g^{h_j}, g^{a_j}, g^{b_j})$.

It sends all attribute authorities' public keys to $\mathbb{A}$ through $\mathbb{B}$.

$\mathbb{A}$ runs the $\text{Setup}_{\text{auth}}$ algorithm to generate the key pairs of the corrupted authorities in the same way.

(3) Query phase I: $\mathbb{B}$ initializes three empty tables $\mathbb{T}$, $\mathbb{T}_1$, $\mathbb{T}_2$, an empty set $\mathbb{D}$, and an integer $j = 0$. Details of queries are described as follows:

(a) Hash query

H_1^* oracle: if the entry (R, MSG, s) already existed in Table $\mathbb{T}_1$, return s. Otherwise, it chooses a random element $s \in \mathbb{Z}_p^*$ (s is unique in Table $\mathbb{T}_1$). Then, it records (R, MSG, s) in Table $\mathbb{T}_1$ and returns s.

H_2^* oracle: if the entry (R, K_{sym}) already existed in Table $\mathbb{T}_2$, return K_{sym}. Otherwise, it chooses a random element $K_{sym} \in \{0,1\}^k$. Then, it records (R, K_{sym}) in Table $\mathbb{T}_2$ and returns K_{sym}.

(b) Key query: In the q-th query, $\mathbb{A}$ queries the decryption key related with an attribute set S_{GID_q} by sending S_{GID_q} and GID_q to $\mathbb{B}$. $\mathbb{B}$ calls $\mathbb{C}$ to generate the decryption key and sends it to $\mathbb{A}$. $\mathbb{C}$ chooses a random number $r \in \mathbb{Z}_p^* \setminus \{-a_j + GID_q/b_j\}$ to compute the decryption key $sk_{GID_q} = (\{sk_{GID_q,j}\}_{j \in S_{AA,GID}}, GID) = (\{K_{1,i}, K_{2,i}, K_{3,i}\}_{i \in S_{GID}}, GID)$ while setting $\mathbb{D} = \mathbb{D} \cup S_{GID_q}$

$$\begin{cases} K_{1,i} = g^{\alpha_i/a_j + GID_q + b_j} H^*(GID_q)^{t_i/a_j + GID_q + b_j}, \\ K_{2,i} = H^*(i)^{h_j}, \\ K_{3,i} = r. \end{cases} \tag{18}$$

$\mathbb{B}$ chooses a random element $a \in \mathbb{Z}_p^*$ to compute $h = g^a$ to simulate the output of the encryption algorithm. $\mathbb{B}$ calls $\mathbb{C}$ to run the outsourced decryption key generation algorithm: $\mathbb{C}$ chooses a random number $z \in \mathbb{Z}_p^*$ to compute

$$\begin{cases} opk_{GID_q} = \left(\left\{K_{1,i}^{1/z}\right\}_{i \in L'}, g^{1/z}, H^*(GID)^{1/z}\right), \\ osk_{GID_q} = z. \end{cases} \tag{19}$$

Sends $ok_{GID_q} = (opk_{GID_q}, osk_{GID_q})$ to $\mathbb{B}$. $\mathbb{B}$ stores the entry $(q, S_{GID_q}, sk_{GID_q}, ok_{GID_q})$ in the table $\mathbb{T}$. Finally, $\mathbb{B}$ returns the key to $\mathbb{A}$.

(c) Decryption query: without loss of generality, we assume that all ciphertexts input to this query have been partially decrypted. For instance, we assume that CT' was correctly decrypted by opk of the entry $(q, S_{GID_q}, sk_{GID_q}, ok_{GID_q})$. Let CT' be associated with a structure $(\mathbb{M}, \rho)$ which is not equal with $(\mathbb{M}^*, \rho^*)$. Let opk be associated with a set of attributes which satisfies $(\mathbb{M}, \rho)$ and not satisfies $(\mathbb{M}^*, \rho^*)$

Table 3: Notations.

Notation	Meaning
E, $/E_T$	One exponentiation in group $\mathbb{G}/\mathbb{G}_T$
P_e	One pairing operation of the pairing function e.
N_e	The row number of the encryption LSSS access matrix.
N_u	The attribute number of the user attribute set.
N_d	The attribute number required in decryption.
N_a	The attribute number of the attribute universe set.

Search Table $\mathbb{T}_1$ to find if there exists an entry (R, MSG, s) which satisfies $(CT')^{osk_{GID_q}} = e(g,g)^s$. If not, abort. Else, obtain entry (R, K_{sym}) in Table $\mathbb{T}_2$. If this entry does not exist, abort. Else, test if $C_0 = Re(g,g)^s$ and $CT_{sym} = Enc_{sym}(K_{sym}, MSG)$. If yes, output MSG. Else, abort.

(4) Challenge: $\mathbb{A}$ chooses two message $MSG_1, MSG_2 \in \{0,1\}^*$ with same length then sends them to $\mathbb{B}$. $\mathbb{B}$ chooses two message $R_0, R_1 \in \mathbb{G}_T$ with same length and then sends them to $\mathbb{C}$. $\mathbb{C}$ chooses a random bit $b \in \{0,1\}$, then $\mathbb{C}$ encrypts R_b under the access structure $(\mathbb{M}^*, \rho^*)$ by running Lewko's scheme. Finally, $\mathbb{C}$ returns $CT_{b,ABE}^*$ to $\mathbb{B}$. $\mathbb{B}$ guesses b with advantage $Adv_A[Exp^{Lewko}]$. Then, $\mathbb{B}$ computes $K_{sym}^* = H_2^*(R_b^*)$ and $CT_{b,sym}^* = Enc_{sym}(K_{sym}^*, MSG_b)$. Finally, $\mathbb{B}$ returns $CT_b^* = (CT_{b,ABE}^*, CT_{b,sym}^*)$ to $\mathbb{A}$

(5) Query phase II: the adversary $\mathbb{A}$ can query a polynomially bounded number of queries as in query phase II after receiving the ciphertext CT_b^* with restrictions that the queried attribute set cannot satisfy the challenge access structure, and the response of the decryption query cannot be either MSG_0 or MSG_1

(6) Guess: $\mathbb{A}$ tries to guess b' based on CT_b^*. Then, $\mathbb{A}$ sends b' to $\mathbb{C}$ through $\mathbb{B}$. If $b' = b$, we say that $\mathbb{A}$ wins this experiment

We can easily get that the advantage of $\mathbb{A}$ to win the experiment Exp^{conf} is smaller than the advantage of $\mathbb{B}$ to win the experiment Exp^{Lewko}, because $\mathbb{A}$ has to be based on the right CT_b^* provided by $\mathbb{B}$ to guess b successfully. In other words, $Pr[Exp_{A^{Lewko}}(1^\xi)] > Pr[Exp_{A^{Conf-Real}}(1^\xi)]$ and our scheme achieve selectively replayable CCA secure.

6.2. Verifiability

Theorem 2. *If H_1 and H_2 are two collision-resistant hash functions, our scheme is verifiable against malicious servers.*

Proof. We define a PPT adversary $\mathbb{A}$ running the experiment defined in Section 4.3(2) with an entity $\mathbb{B}$. $\mathbb{B}$ tries to break the collision resistance of the two hash functions H_1^* and H_2^*.

(1) Initialization: the adversary $\mathbb{A}$ submits a challenge access policy $\Psi^* = (\mathbb{M}^*, \rho^*)$ to the entity $\mathbb{B}$

Table 4: Storage cost comparison.

Scheme	[11]	[13]	[14]	Our scheme
Decryption key length	$2N_u \mid \mathbb{G} \mid$	$3N_u \mid \mathbb{G} \mid + N_u \mid \mathbb{Z}_p^* \mid + \mid \mathbb{Z}_p \mid$	$(2N_u + 3) \mid \mathbb{G} \mid + 2 \mid \mathbb{Z}_p \mid$	$2N_u \mid \mathbb{G} \mid + N_u \mid \mathbb{Z}_p \mid$
Outdec key length	$(N_d + 2) \mid \mathbb{G} \mid + \mid \mathbb{Z}_p^* \mid$	—	$(2N_u + 2) \mid \mathbb{G} \mid$	$(N_d + 2) \mid \mathbb{G} \mid + \mid \mathbb{Z}_p^* \mid$
Ciphertext length	$(3N_e + 1) \mid \mathbb{G} \mid + 1 \mid \mathbb{G}_T \mid$	$5N_e \mid \mathbb{G} \mid + (N_e + 1) \mid \mathbb{G}_T \mid$	$(3N_e + 2) \mid \mathbb{G} \mid + 1 \mid \mathbb{G}_T \mid$	$(5N_e + 1) \mid \mathbb{G} \mid + 1 \mid \mathbb{G}_T \mid$

(2) Set-up: $\mathbb{B}$ runs the Setup algorithm to generate the global parameter except the two hash functions

(3) $\mathbb{B}$ runs the $\text{Setup}_{\text{auth}}$ algorithm to generate keypairs of attribute authorities.

Query: $\mathbb{A}$ runs the adversary queries as defined in query phase I and query phase II through $\mathbb{B}$ to get the related decryption keys and outsourced decryption keys

(4) Challenge: $\mathbb{A}$ sends the challenge message MSG^* to $\mathbb{B}$, and $\mathbb{B}$ answers as follows

It chooses a random message $\text{R}^* \in \text{G}_\text{T}$ to run Lewko's encryption scheme to encrypt R^* under the access policy $(\mathbb{M}^*, \rho^*)$.

It computes $s^* = H_1^*(R^* \| \text{MSG}^*)$ and $K_{\text{sym}}^* = H_2^*(R^*)$.

It runs the symmetric encrypt algorithm to encrypt MSG* to generate the ciphertext $\text{CT}^* = (\text{CT}_{\text{ABE}}^*, \text{CT}_{\text{sym}}^*) = (\text{h}, \text{C}_0, \{\text{C}_{1,\text{i}}, \text{C}_{2,\text{i}}, \text{C}_{3,\text{i}}, \text{C}_{4,\text{i}}, \text{C}_{5,\text{i}}\}, \text{CT}_{\text{sym}}^*)$.

It returns the ciphertext $\text{CT}^* = (\text{CT}_{\text{ABE}}^*, \text{CT}_{\text{sym}}^*)$ to $\mathbb{A}$.

If $\mathbb{B}$ can recover a message $\text{MSG} \in \{\text{MSG}^*, \perp\}$, then we say $\mathbb{A}$ wins this experiment. Hence there are two cases are considered:

(1) $(\text{MSG}, \text{R}) \neq (\text{MSG}^*, \text{R}^*)$, which means that $\mathbb{B}$ finds a collision of the hash function H_1^*

(2) $(\text{K}_{\text{sym}}, \text{CT}_{\text{sym}}) = (\text{K}_{\text{sym}}^*, \text{CT}_{\text{sym}}^*)$ but $(\text{R}^* \neq \text{R})$, which means that $\mathbb{B}$ breaks the collision resistance condition of H_2^* such as $H_2^*(\text{R}) = K_{\text{sym}} = K_{\text{sym}}^* = H_2^*(R^*)$

In other words, since H_1 and H_2 are two collision-resistant hash functions, the outsourced decryption of our scheme is verifiable.

6.3. Fully Hiding

Theorem 3. *Our scheme is an outsourced ABE with fully hidden policy if the one-way anonymous key agreement protocol [15] is IND-CPA secure.*

Proof. The purpose of this proof is that no PPT adversary can recover the access policy without the right decryption key. The setup phase and the query phase 1 are same as the confidentiality experiment.

In the challenge phase, the adversary $\mathbb{A}$ chooses two challenge messages R_0^*, R_1^* and two valid access policies Ψ_0, Ψ_1, and then it sends them to the challenger $\mathbb{C}$. Notice, Ψ_0 and Ψ_1 satisfy the following restriction: either all the attribute sets queried in query phase 1 satisfy none of the policy or all attribute sets satisfy both the policies. Then, $\mathbb{C}$ computes $q'(i) = e((g^{h_j})^a, H(i))$ based on the one-way anonymous key agreement protocol where $a \in \mathbb{Z}_p^*$ is a random number. This step is used to hide the real policy by replacing each attributes in the policy with the corresponding $q'(i)$. Then, $\mathbb{C}$ chooses a random bit $b \in \{0, 1\}$ and encrypts the message R_b^* under the access policy Ψ_b. Finally, $\mathbb{C}$ sends CT^* to $\mathbb{A}$. After that, $\mathbb{A}$ still can query a polynomially bounded number of queries as in query phase I. The none-or-both principle still works in this phase.

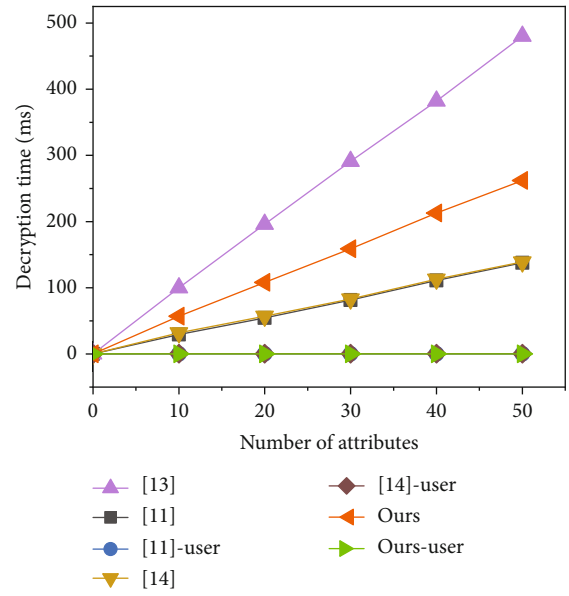

Figure 7: Decryption cost.

In the guess phase, $\mathbb{A}$ outputs b'.

When $\mathbb{A}$ tries to decrypt CT^*, it has to recover the access policy first. In our scheme, the decryption key $\text{K}_{2,i} = H(i)^{h_j}$ is necessary for it to compute $q'(i)$ because we computed the $\text{q}'(i)$ based on the one-way anonymous key agreement protocol before we encrypted the message. It means only the authorized user can get the right access policy. And due to the random value a, unauthorized user cannot guess attribute i from $q'(i)$ which prevents the collusion of the users. Hence, the advantage of the adversary to win the experiment $\text{Adv}_A[\text{Exp}^{\text{Priv}}(1^\xi)]$ is negligible, and our scheme ensures the privacy preservation of the access policy against adaptive chosen plaintext attack.

6.4. Traceability. In this section, we prove that our scheme is fully traceable under the q-SDH assumption.

Lemma 1. *Our scheme achieves fully user traceability based on that the Boneh-Boyen fully signature scheme [31] is strong existential forgery secure against adaptive chosen message attack.*

Proof. We define a PPT adversary $\mathbb{A}$ running the experiment defined in Section 4.3(4) to attack our scheme through an entity $\mathbb{B}$ by $\mathbb{B}$ breaking the Boneh-Boyen fully signature scheme with the same advantage under adaptive chosen message attacks. Assuming the advantage of the adversary $\mathbb{A}$ to break our scheme is ε, and $\mathbb{B}$ can access a random oracle H. Let $\mathbb{C}$ be the challenger in the B-B scheme, $Sig_j \in \mathbb{G}$ be the signature of AA_j, and $pk_{AA_j}^{sig} = \{\mathbb{G}, p, g, g^{a_j}, g^{b_j}\}$ is the associated public key of Sig_j.

(1) Set-up: the challenger $\mathbb{C}$ runs the $Setup$ algorithm to generate the global parameter PP and sends PP to $\mathbb{B}$. For each noncorrupted authorities in the set S', $\mathbb{C}$ sends pk_{AAj}^{sig} to $\mathbb{B}$. Then, $\mathbb{B}$ chooses two random numbers α_i, β_i for each attribute in the attribute set of the authority, and then $\mathbb{B}$ chooses a random number h_j to generate the public key of the authority $pk_{\mathrm{AA}j} = (\{g^{\alpha_i}, g^{\beta_i}\}_{i \in S_{\mathrm{A,AA}_j}}, g^{h_j}, g^{a_j}, g^{b_j})$. Finally, $\mathbb{B}$ returns PP and $\{pk_{\mathrm{AA}j}\}_{j \in S'}$ to $\mathbb{A}$. For corrupted authorities, $\mathbb{A}$ runs the $Setup_{\mathrm{auth}}$ algorithm to generate the key pairs for them

(2) Key query: $\mathbb{A}$ runs m queries. In the q-th query, $\mathbb{A}$ sends $(S_{\mathrm{GID}_q}, \mathrm{GID}_q)$ to $\mathbb{B}$. $\mathbb{B}$ initiates an empty table $\mathbb{T}$ and do the following steps

(a) Accesses the random oracle $H(\mathrm{GID}_q)$: $\mathbb{B}$ searches the entry $(\mathrm{GID_q}, t_{\mathrm{GID}_q}, g^{t_{\mathrm{GID}_q}})$ in the table $\mathbb{T}$, and if it exists, $\mathbb{B}$ outputs $\mathrm{g}^{t_{\mathrm{GID}_q}}$. Else, $\mathbb{B}$ chooses a random number $\mathrm{t}_{\mathrm{GID}_q}$ while stores $(\mathrm{GID}_q, t_{\mathrm{GID}_q}, g^{t_{\mathrm{GID}_q}})$ in the table $\mathbb{T}$. $\mathbb{B}$ outputs $\mathrm{g}^{t_{\mathrm{GID}_q}}$

(b) Generates the decryption key $sk_{\mathrm{GID}_{q,j}}$: $\mathbb{C}$ chooses a random number $\mathrm{r} \in \mathbb{Z}_p^* \setminus \{-a_j + \mathrm{GID}_q/b_j\}$ for each attribute $\mathrm{i} \in S_{\mathrm{GID}_q,j}$ and returns the signature $(r, \sigma = g^{1/a_j+GID_q+b_jr})$. Then, $\mathbb{B}$ computes the components of $sk_{\mathrm{GID}_q,j}$

$$\begin{cases} K_{1,i} = \sigma^{\left(\alpha_i+\beta_i t_{GID_q}\right)} = g^{\alpha_i+\beta_i t_{GID_q}/a_j+GID_q+b_jr} = g^{\alpha_i/a_j+GID_q+b_jr} g^{t_{GID_q}\beta_i/a_j+GID_q+b_jr=g^{\alpha_i/a_j+GID_q+b_jr}} H\left(GID_q\right)^{\beta_i/a_j+GID_q+b_jr}, \\ K_{2,i} = H(i)^{h_j}, \\ K_{3,i} = r. \end{cases} \tag{20}$$

Then, $\mathbb{B}$ sets $\mathbb{D} = \mathbb{D} \cup S_{\mathrm{GID}_q}$. Finally $\mathbb{B}$ returns the following result to $\mathbb{A}$:

$$sk_{GID_q} = \left(\left\{sk_{GID_q,j}\right\}_{j \in S_{AA,GID}}, GID_q\right) = \left(\left\{K_{1,i}, K_{2,i}, K_{3,i}\right\}_{i \in S_{GID_q}}, GID_q\right). \tag{21}$$

(3) Key forgery: $\mathbb{A}$ sends a sk^* to $\mathbb{B}$. The advantage of the adversary to win is defined as

$$\Pr\left[\mathrm{Trace}\left(PP, \left\{pk_{\mathrm{AAj}}\right\}, \mathrm{sk}^*\right) \in \{\perp, GID_1, \cdots, GID_m\}\right] = \varepsilon, \tag{22}$$

where $(\mathrm{GID}_1, \cdots, \mathrm{GID}_m)$ is mGID queried in the last phase. If $\mathrm{Trace}(\mathrm{PP}, \{pk_{\mathrm{AAj}}\}, sk^*) \in \{\perp, \mathrm{GID}_1, \cdots, \mathrm{GID}_m\}$, it means $sk^* = (\{K_{1,i}, K_{2,i}, K_{3,i}\}_{i \in \mathrm{S_{GID}}}, \mathrm{GID})$ passed the form check and $\mathrm{GID} \in \{\perp, \mathrm{GID}_1, \cdots, \mathrm{GID}_m\}$. Hence, $\exists i \in S$, s.t.

$$\mathrm{K}_{1,i}, K_{2,i} \in \mathbb{G}, K_{3,i}, \mathrm{GID} \in \mathbb{Z}_{p,}^*$$
$$\mathrm{e}\left(K_{1,i}, g^{a_j} g^{\left(b_j\right)^{K_{3,i}}} g^{GID}\right) = e(g, g)^{\alpha_i} e\left(H(\mathrm{GID}), g^{t_i}\right). \tag{23}$$

Without loss of generality, we assume the adversary $\mathbb{A}$ accessed the random oracle $H(\mathrm{GID})$ before it outputs the sk^*. $\mathbb{B}$ obtains the entry $(\mathrm{GID}, t_{\mathrm{GID}}, g^{t_{\mathrm{GID}}})$ from the table $\mathbb{T}$. According to $\mathrm{e}(K_{1,i}, g^{a_j} g^{(b_j)^{K_{3,i}}} g^{\mathrm{GID}}) = e(g, \mathrm{g})^{\alpha_i} e(H(\mathrm{GID}), g^{\beta_i})$, we can get $\mathrm{K}_{1,i} = g^{\alpha_i+t_{\mathrm{GID}}\beta_i/a_j+b_jK_{3,i}+\mathrm{GID}}$. Then, $\mathbb{B}$ computes the signature $\sigma_j = (K_{1,i})^{1/\alpha_i+t_{\mathrm{GID}}\beta_i}$. Because $GID, K_{3,i} \in \mathbb{Z}_p^*$, hence $(K_{3,i}, \sigma_j)$ is a valid signature on message GID in the B-B signature scheme. Because $GID \in \{\mathrm{GID}_1, \cdots, \mathrm{GID}_m\}$, it means $\mathbb{B}$ never queried the signature of GID before, and the advantage of $\mathbb{B}$ to break the B-B scheme is equal with the advantage of the adversary $\mathbb{A}$ to break our scheme, which is ε.

Table 5: Computational cost comparison.

Scheme	[11]	[13]	[14]	Our scheme
Key generation	$3N_uE$	$5N_uE$	$(4N_u + 4)E$	$3N_uE$
Encryption	$(N_u + 5N_e + 1)E + 1E_T + N_uP_e$	$(2N_e + 1)E_T + 6N_eE$	$(5N_e + 2)E + 1E_T$	$(N_u + 7N_e + 1)E + 1E_T + N_uP_e$
Outdecryption	$N_dE_T + 3N_dP_e$	—	$N_dE_T + (3N_d + 1)P_e$	$3N_dE_T + 3N_dP_e$
User decryption	$1E_T$	$4N_dE_T + 3N_dP_e$	$3E_T$	$1E_T$

According to the Boneh-Boyen signature scheme, we can also get the following lemma.

Lemma 2. *If the q-SDH assumption holds in the group $\mathbb{G}$, the full signature scheme of Boneh and Boyen is strong existential forgery secure against adaptive chosen message attacks.*

Theorem 4. *If the q-SDH assumption holds in the group $\mathbb{G}$, our scheme achieves fully user traceability.*

Proof. It follows directly from the above Lemma 1 and Lemma 2.

7. Performance Analysis

The notations used in our performance analysis are summarized in Table 3.

The comparison of storage cost and computational cost between our scheme and some other ABE schemes is illustrated in Tables 4 and 5separately. Notice that all results do not contain the costs of the symmetric cryptography including hash operations.

From Table 4, we can see that the decryption key lengths of scheme [11] and ours are related to the number of attributes used in decryption as both scheme outsource the most decryption work to the cloud server, while the decryption key length of scheme [14] is related to the number of attributes in user attribute sets. Speaking of the length of the ciphertext, of all four schemes are associated with the row number of the encryption LSSS access matrix.

As we can see from Table 4, scheme [13] needs $5N_u$ exponentiations in group $\mathbb{G}$ to generate the user decryption key. It needs $2N_e + 1$ exponentiations in group $\mathbb{G}_T$ and $6N_e$ exponentiations in group $\mathbb{G}$ in the encryption phase. Specially, 4 N_d exponentiations in group $\mathbb{G}_T$ and $3N_d$ pairings are costed by a user who needs to decrypt in scheme [13], which is too heavy for resource-limited IoT devices. Scheme [11] is an ABE scheme with outsourced decryption which needs $3N_u$ exponentiations in group $\mathbb{G}$ in the key generation phase. It requires $N_u + 5N_e + 1$ exponentiations in group $\mathbb{G}$, one exponentiation in group $\mathbb{G}_T$, and N_u pairings to encrypt. As the most pairing operations are done by the cloud server, users only cost one exponentiation in group $\mathbb{G}_T$ to decrypt in [11].

Li et al. proposed a traceable ABE scheme which needs $4N_u + 4$ exponentiations in group $\mathbb{G}$ to generate the private key [14]. In encryption phase, users spends $5N_e + 2$ exponentiations in group $\mathbb{G}_T$ and one exponentiation in group $\mathbb{G}$, while the cloud server performs N_d exponentiations in group $\mathbb{G}_T$ as well as $3N_d + 1$ pairings to predecrypt in [14]. As a result, users only cost three exponentiations in group $\mathbb{G}_T$ to fully decrypt.

Our scheme needs $3N_u$ exponentiations in group $\mathbb{G}$ in the key generation phase. To achieve fully policy hidden which is deeply valuable in some healthy data application, our scheme requires $N_u + 7N_e + 1$ exponentiations in group $\mathbb{G}$, one exponentiation in group $\mathbb{G}_T$, and N_u pairings to encrypt. Meanwhile, our scheme realizes verifiable outsourced decryption. Our scheme outsources $3N_d$ exponentiations in group $\mathbb{G}_T$ and $3N_d$ pairings to the cloud server. Thus, IoT devices in our scheme only require one exponentiation in group $\mathbb{G}_T$ to decrypt, which dramaticlly reduces the computational overhead of resource-limited devices.

Figure 7 illustrates the time overhead of decryption. The simulation is performed in a Ubuntu 16.4 desktop system with 3.0-GHz Intel Core (TM) i5-7400 CPU and 2-GB RAM, and all experiments are done by using the Charm (version 0.50) [37], a rapid prototyping framework for cryptographic schemes based with Python.

Compared with the outsourced multiauthority ABE scheme [11] with no traceability, our traceable MAABE scheme is with little extra computational cost. However, the user decryption cost of [11] and our scheme is same owing to the outsourced decryption. While comparing with the traceable single-authority ABE scheme [14], our multiauthority scheme can handle more attributes and is more suitable for a large number of devices of IoT systems. In addition, another traceable MAABE [13] is not applicable for resource-limited IoT devices due to its heavy decryption cost.

More importantly, our scheme costs barely one hash operation to achieve the verification of decryption results. About another practical function is achieved by our scheme, traceability, and the cost of our scheme is $N_a(2E + H + 3P)$. Although it looks like that this result is linear to the size of the attribute universe set, the real computational cost of this algorithm for each AA is linear to the size of its own attribute set as we assumed that attribute sets controlled by different attribute authorities are disjoint in our scheme.

8. Conclusion

In this paper, we propose a multiauthority ABE scheme supporting verifiable outsourced decryption and white-box traceability. Our scheme outsources most decryption works to the honest-but-curious resource-rich cloud server; thus, our scheme meets the special needs of resource-limited IoT devices. Moreover, our scheme protects the privacy of both the encryptor and the decryptor by the fully hiding policy technology. At the same time, another issue influences the application of ABE—the key leakage problem—which is

solved by the user traceability algorithm. In a word, our scheme realizes several practical functions while achieving replayable chosen-ciphertext attack security.

In the future, we plan to improve the scheme with fixed key size and ciphertext size to further reduce equipment overheads. Moreover, we can also consider how to solve another difficulty of the practical application of the ABE—attribute revocation and user revocation. How to dynamically withdraw attributes or users without affecting other authorized users is the focus of our future works.

Data Availability

All data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Key Research and Development Program of China under Grant 2019YFB2102600; in part by the National Natural Science Foundation of China under Grants 62072065, 61832012, 61672321, 61771289, and 61373027; in part by the Fundamental Research Funds for the Central Universities under Grant 2019CDQYRJ006; in part by the Chongqing Research Program of Basic Research and Frontier Technology under Grant cstc2018jcyjAX0334; in part by the Key Project of Technology Innovation and Application Development of Chongqing under Grant CSTC2019jscx-mbdx0151; and in part by the Overseas Returnees Innovation and Entrepreneurship Support Program of Chongqing under Grants cx2018015 and cx2020004.

References

[1] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Workshop on the theory and application of cryptographic techniques*, pp. 47–53, Springer, 1984.

[2] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *Annual international cryptology conference*, pp. 213–229, Springer, 2001.

[3] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 457–473, Springer, 2005.

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

[5] S. Pattar, R. Buyya, K. R. Venugopal, S. S. Iyengar, and L. M. Patnaik, "Searching for the iot resources: fundamentals, requirements, comprehensive review, and future directions," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 3, pp. 2101–2132, 2018.

[6] M. Chase, "Multi-authority attribute based encryption," in *Theory of Cryptography Conference*, pp. 515–534, Springer, 2007.

[7] M. Chase and S. S. M. Chow, "Improving privacy and security in multi-authority attribute-based encryption," in *Proceedings of the 16th ACM conference on Computer and communications security - CCS '09*, pp. 121–130, Chicago Illinois USA, 2009.

[8] A. Lewko and B. Waters, "Decentralizing attribute-based encryption," in *Annual international conference on the theory and applications of cryptographic techniques*, pp. 568–588, Springer, Tallinn, Estonia, 2011.

[9] M. Green, S. Hohenberger, and B. Waters, "Outsourcing the decryption of abe ciphertexts," in *Proc. 20th USENIX Security Symposium, USENIX Association*, vol. 2011, pp. 1–16, San Francisco, CA, 2011.

[10] J. Li, F. Sha, Y. Zhang, X. Huang, and J. Shen, "Verifiable outsourced decryption of attribute-based encryption with constant ciphertext length," *Security and Communication Networks*, vol. 2017, Article ID 3596205, 11 pages, 2017.

[11] S. Belguith, N. Kaaniche, M. Laurent, A. Jemai, and R. Attia, "Phoabe: securely outsourcing multi-authority attribute based encryption with policy hidden for cloud assisted iot," *Computer Networks*, vol. 133, pp. 141–156, 2018.

[12] Z. Liu, S. Duan, P. Zhou, and B. Wang, "Traceable-then-revocable ciphertext-policy attribute-based encryption scheme," *Future Generation Computer Systems*, vol. 93, pp. 903–913, 2019.

[13] K. Zhang, H. Li, J. Ma, and X. Liu, "Efficient large-universe multi-authority ciphertext-policy attribute-based encryption with white-box traceability," *Science China Information Sciences*, vol. 61, no. 3, article 032102, 2018.

[14] Q. Li, H. Zhu, Z. Ying, and T. Zhang, "Traceable ciphertext-policy attribute-based encryption with verifiable outsourced decryption in eHealth cloud," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 1701675, 12 pages, 2018.

[15] H. Zhong, W. Zhu, Y. Xu, and J. Cui, "Multi-authority attribute-based encryption access control scheme with policy hidden for cloud storage," *Soft Computing*, vol. 22, no. 1, pp. 243–251, 2018.

[16] S. Belguith, N. Kaaniche, A. Jemai, M. Laurent, and R. Attia, "Pabac: a privacy preserving attribute based framework for fine grained access control in clouds," in *13th IEEE International Conference on Security and Cryptography(Secrypt)*, pp. 133–146, Portugal, 2016.

[17] Z. Cai, Z. He, X. Guan, and Y. Li, "Collective data-sanitization for preventing sensitive information inference attacks in social networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 4, pp. 577–590, 2018.

[18] V. Božović, D. Socek, R. Steinwandt, and V. I. Villányi, "Multi-authority attribute-based encryption with honest-but-curious central authority," *International Journal of Computer Mathematics*, vol. 89, no. 3, pp. 268–283, 2012.

[19] Z. Cai and X. Zheng, "A private and efficient mechanism for data uploading in smart cyber-physical systems," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 2, pp. 766–775, 2020.

[20] C. Hu, N. Zhang, H. Li, X. Cheng, and X. Liao, "Body area network security: a fuzzy attribute-based signcryption scheme," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 9, pp. 37–46, 2013.

[21] X. Zheng, Z. Cai, J. Yu, C. Wang, and Y. Li, "Follow but no track: privacy preserved profile publishing in cyber-physical social systems," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 1868–1878, 2017.

[22] Y. Pu, C. Hu, S. Deng, and A. Alrawais, "R²PEDS: a recoverable and revocable privacy-preserving edge data sharing scheme," *IEEE Internet of Things Journal*, vol. 7, no. 9, pp. 8077–8089, 2020.

[23] Z. Cai and Z. He, "Trading private range counting over big iot data," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pp. 144–153, Dallas, TX, USA, July 2019.

[24] X. Xu, J. Zhou, X. Wang, and Y. Zhang, "Multi-authority proxy re-encryption based on cpabe for cloud storage systems," *Journal of Systems Engineering and Electronics*, vol. 27, no. 1, pp. 211–223, 2016.

[25] Y. Yang, H. Zhu, H. Lu, J. Weng, Y. Zhang, and K.-K. R. Choo, "Cloud based data sharing with fine-grained proxy re-encryption," *Pervasive and Mobile Computing*, vol. 28, pp. 122–134, 2016.

[26] X. Zheng, Z. Cai, and Y. Li, "Data linkage in smart internet of things systems: a consideration from a privacy perspective," *IEEE Communications Magazine*, vol. 56, no. 9, pp. 55–61, 2018.

[27] J. Lai, R. H. Deng, C. Guan, and J. Weng, "Attribute-based encryption with verifiable outsourced decryption," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 8, pp. 1343–1354, 2013.

[28] N. Deng, S. Deng, C. Hu, and K. Lei, "An efficient revocable attribute-based signcryption scheme with outsourced designcryption in cloud computing," in *International Conference on Wireless Algorithms, Systems, and Applications*, pp. 84–97, Springer, 2019.

[29] T. Nishide, K. Yoneyama, and K. Ohta, "Attribute-based encryption with partially hidden encryptor-specified access structures," in *International conference on applied cryptography and network security*, pp. 111–129, Springer, 2008.

[30] M. J. Hinek, S. Jiang, R. S. Naini, and S. F. Shahandashti, "Attribute-based encryption without key cloning," *International Journal of Applied Cryptography*, vol. 2, no. 3, pp. 250–270, 2012.

[31] D. Boneh and X. Boyen, "Short signatures without random oracles," in *International conference on the theory and applications of cryptographic techniques*, pp. 56–73, Springer, 2004.

[32] Zhen Liu, Zhenfu 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.

[33] Z. Liu, Z. Cao, and D. S. Wong, "Traceable cp-abe: how to trace decryption devices found in the wild," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 1, pp. 55–68, 2015.

[34] G. Yu, Y. Wang, Z. Cao, J. Lin, and X. Wang, "Traceable and undeniable ciphertext-policy attribute-based encryption for cloud storage service," *International Journal of Distributed Sensor Networks*, vol. 15, no. 4, 2019.

[35] H. Qiao, J. Ren, Z. Wang, H. Ba, and H. Zhou, "Compulsory traceable ciphertext-policy attribute-based encryption against privilege abuse in fog computing," *Future Generation Computer Systems*, vol. 88, pp. 107–116, 2018.

[36] R. Canetti, H. Krawczyk, and J. B. Nielsen, "Relaxing chosen-ciphertext security," in *Annual International Cryptology Conference*, pp. 565–582, Springer, 2003.

[37] J. A. Akinyele, C. Garman, I. Miers et al., "Charm: a framework for rapidly prototyping cryptosystems," *Journal of Cryptographic Engineering*, vol. 3, no. 2, pp. 111–128, 2013.