Int. J. Inf. Secur.
DOI 10.1007/s10207-014-0270-9

SPECIAL ISSUE PAPER

Privacy-preserving personal health record using multi-authority attribute-based encryption with revocation

Huiling Qian · Jiguo Li · Yichen Zhang · Jinguang Han

Abstract Personal health record (PHR) service is an emerging model for health information exchange. In PHR systems, patient's health records and information are maintained by the patient himself through the Web. In reality, PHRs are often outsourced to be stored at the third parties like cloud service providers. However, there have been serious privacy concerns about cloud service as it may expose user's sensitive data like PHRs to those cloud service providers or unauthorized users. Using attribute-based encryption (ABE) to encrypt patient's PHRs in cloud environment, secure and flexible access control can be achieved. Yet, problems like scalability in key management, fine-grained access control, and efficient user revocation remain to be addressed. In this paper, we propose a privacy-preserving PHR, which supports fine-grained access control and efficient revocation. To be specific, our scheme achieves the goals (1) scalable and fine-grained access control for PHRs by using multi-authority ABE scheme, and (2) efficient on-demand user/attribute revocation and dynamic policy update. In our scheme, we consider the situation that multiple data owners exist, and patient's PHRs are encrypted and stored in semi-trust servers. The access structure in our scheme is expressive access tree structure, and the security of our scheme can be reduced to the standard decisional bilinear Diffie–Hellman assumption.

Keywords Personal health records · Cloud computing · Data privacy · Fine-grained access control · Attribute-based encryption

H. Qian · J. Li (✉) · Y. Zhang
College of Computer and Information, Hohai University,
Nanjing 211100, Jiangsu, China
e-mail: ljg1688@163.com; lijiguo@hhu.edu.cn

H. Qian
e-mail: jsntqhl@126.com

Y. Zhang
e-mail: zyc_718@163.com

J. Han
Jiangsu Provincial Key Laboratory of E-Business,
Nanjing University of Finance and Economics,
Nanjing 210003, Jiangsu, China
e-mail: jghan22@gmail.com

1 Introduction

PHR service is a simple health record storage service, which provides a complete summary of patient's medical history online rather than on paper, and allows a patient to create, manage, control, and share his PHRs with a wide range of users, including family members, friends, doctors, and health care providers. Furthermore, PHR systems based on Web can be easily integrated with other services such as mobile applications, thus making the management of PHRs more convenient and efficient for patients. Recently, with the development of cloud computing, PHRs are often outsourced to be stored at the third parties like cloud service providers, which results in security issues [1,2]. Some architectures that store PHRs in cloud computing are proposed in [3,4]. However, when outsourcing PHRs to cloud service providers, patients will lose the direct control of hardwares that store patients' PHRs. PHRs stored in cloud service providers will suffer from more attacks both from outside and inside than paper-based PHR. Besides height, weight, and other information about patient's body, PHRs also include some very sensitive data like disease. Moreover, laws like the health insurance portability and accountability act (HIPAA) [5] require patient's privacy information to be protected. Therefore, to provide secure and privacy-preserving PHR system with fine-grained access control is essential.

A promising way to realize secure and privacy-preserving PHR system is to encrypt the data before outsourcing it to the cloud. Patient should be able to share his PHRs with his friends and decide what kind of users can access what kind of PHRs. Only users with corresponding secret keys can access the corresponding encrypted PHRs. The concept of attribute-based encryption (ABE) first introduced by Sahai and Waters [6] can exactly be used to encrypt PHRs in PHR systems. In ABE schemes, users can decrypt the message when their attributes satisfy the specified access structure. ABE schemes can be divided into two categories, namely ciphertext-policy attribute-based encryption (CP-ABE) [7–10] and key-policy attribute-based encryption (KP-ABE) [11,12]. In KP-ABE systems, user's secret keys are associated with an access structure that is determined by the trusted authority, while the ciphertext is labeled with a set of attributes. In CP-ABE systems, the user's secret keys are labeled with a set of attributes, while the ciphertext is associated with an access structure that is determined by the encryptor. In PHR systems, a patient can share his PHRs by specifying an access structure in ABE schemes. Furthermore, patients should also have the right to revoke user/attribute and update policies when it is necessary [13].

In PHR systems, there are two kinds of authorized users. For users like families and friends, they access the PHRs for personal use. For users like doctors and researchers, they access the PHRs for professional purpose. In paper [4], authors divided the users in the system into two types of domains, namely *personal domains* (PSDs) and *public domains* (PUDs). In PSDs, only limited users like families and friends are involved and a KP-ABE scheme can be used to encrypt PHRs. In this situation, patients work as the authority and issue secret keys associated with an access policy, which defines user's access ability. Ciphertext is labeled with a set of attributes. In PUDs, a larger scale of users including doctors and researchers are involved. In this situation, a multi-authority CP-ABE scheme can be used to encrypt PHRs. In this paper, we mainly focus on PUDs.

Motivations and contributions A PHR system may involve quite a lot of organizations or domains. For example, a patient will need health care domains to process his health information and insurance domains to process his insurance information. Therefore, single-authority ABE schemes will not be sufficient to realize a PHR system. Moreover, let a patient work as an authority to grant access rights to users like doctors and researchers are not realistic since this kind of users has potentially large scale. However, using multi-authority ABE schemes to encrypt PHRs may exist privacy issues. In many existing multi-authority ABE schemes, user's secret key has to be tied to a global identifier GID in order to resist user collusion attacks. Corrupted authorities can cooperatively trace the user via his GID and impersonate the user by collecting his attributes. In this paper, we solve the above problems and propose a multi-authority CP-ABE scheme that can be used to realize privacy-preserving PHR system. To protect user's privacy, we use an anonymous key issuing protocol to generate user's secret key. Thus, authorities can get nothing about user's GID and therefore cannot collect user's attributes by tracing user's GID.

In a PHR system, when a doctor gets promoted or leaves the system, his attributes should be changed correspondingly. Therefore, we require a PHR system allow user/attribute revocation. In this paper, an efficient and on-demand user/attribute revocation is also achieved. Furthermore, our scheme also allows policy update when patient wants to change the access policy associated with ciphertext.

1.1 Related work

In this section, we discuss the works that relate to this paper.

Multi-authority ABE The first multi-authority ABE scheme was proposed by Chase [14] in 2007. In this scheme, there are multiple authorities to distribute secret keys and monitor attributes. One special authority is called central authority, which is responsible for generating public and secret keys for other authorities. User gets his secret keys from multiple authorities. Different from single-authority ABE, it is very hard to resist user collusion attacks in multi-authority ABE schemes. Chase [14] solved the problem by introducing global identifier GID. Each user has a unique GID. User's secret keys must be tied to the GID. Thus, even if malicious users collude, they cannot decrypt a ciphertext that none of them alone can decrypt. Although the problem of user collusion attack is solved, it compromises user's privacy. Multiple malicious authorities can collaborate to collect user's attributes by tracing user's GID.

Lin et al. [15] constructed a multi-authority ABE scheme without a trusted central authority. In this scheme, multiple authorities must work together to initialize the system in the setup stage. Moreover, to add an attribute in the system, multiple authorities must work together to reset the system. Chase and Chow [16] proposed another multi-authority ABE scheme without a trusted central authority. They use a distributed pseudorandom functions (PRF) to remove the trusted central authority. Notably, in this scheme, authors considered the problem of user's privacy. By using a anonymous key issuing protocol, user can finally get his secret key and authority can get nothing about user's GID, thus gets no privacy information about the user. In this paper, we use the technique PRF to construct a privacy-preserving multi-authority CP-ABE scheme.

Revocable ABE ABE is a promising cryptographic primitives for secure and fine-grained access control. However, before deploying ABE in reality, revoking user/attribute effectively and on-demand is a most challenging problem that needs to be solved. In traditional revocable work [17–19], the authority periodically updates secret key to achieve user/attribute revocation. These schemes do not support backward and forward secrecy. Recently, CP-ABE schemes with immediate attribute revocation are proposed in [20–22]. However, all these schemes do not support multi-authority ABE.

Li et al. [4] combined the work in [23] and [16], and proposed a revocable multi-authority ABE scheme. In this paper, we use the similar technique that used in [4] and construct a multi-authority CP-ABE scheme that supports immediate and fine-grained lazy attribute revocation.

ABE for fine-grained access control Fine-grained access control allows flexibility in specifying access rights for a user. Goyal et al. [11] deployed access tree structure into KP-ABE scheme to realize fine-grained access control. Other works that realize fine-grained access control for outsourced data can be found in [18,20,23,24]. Recently, some ABE schemes were applied to electronic health care record (EHR) and PHR system for fine-grained access control. A CP-ABE variant scheme [25] enables secure storage and data sharing of PHR system. In [26], by applying ABE, a self-protecting electronic medical record (EMR) is designed. In this system, EMRs can either be stored on the server or stored on devices like cell phones. However, both [25,26] are designed for single-authority situation. This is not in line with reality and may exist problems like key escrow. Li et al. [4] proposed a PHR system using multi-authority ABE. In this paper, we construct a multi-authority ABE scheme for fine-grained access, which can be used in PHR system for multi-authority situation.

1.2 Organization

We organize the rest of the paper as follows. In Sect. 2, we review bilinear maps, related complexity assumption, and some definitions including access structure, formal definition of N-authority CP-ABE, and security model. In Sect. 3, we present an anonymous key issuing protocol, which is used to achieve user privacy against malicious authorities. In Sect. 4, we present our construction, and in Sect. 5, we provide security proofs and comparisons with existing schemes. Finally, we conclude the paper in Sect. 6.

2 Preliminaries

2.1 Definitions

We first give the definition of access structure and then give the formal definitions of N-authority CP-ABE and security model. In these definitions, attributes will be used to describe users and access structure will be associated with ciphertext.

Definition 1 (*Access structure* [27]) Let P be a set of parties and $P = \{P_1, P_2, \ldots, P_n\}$. A collection $\mathbb{A} \subseteq 2^P$ is considered to be monotone if $\forall B, C$: if $B \in \mathbb{A}$ and $B \subseteq C$, then $C \in \mathbb{A}$. An access structure (respectively, monotonic access structure) is a collection (respectively, monotone collection) $\mathbb{A}$ of non-empty subsets of $\{P_1, P_2, \ldots, P_n\}$, namely $\mathbb{A} \subseteq 2^P \setminus \{\emptyset\}$. The sets in $\mathbb{A}$ are called the authorized sets, and the sets not in $\mathbb{A}$ are called the unauthorized sets.

In this paper, we will use an access tree structure, which is also used in [7,11,28] to specify what kind of conditions users' secret keys should satisfy to decrypt ciphertexts. In an access tree structure, each interior node is a threshold gate, and leaves are associated with attributes. Access tree structure is very expressive since each threshold gate in interior nodes can be "AND" or "OR" gates. A user can successfully decrypt a ciphertext if and only if the attributes that used to label user's secret key satisfy the access tree associated with ciphertext. In this paper, we use the same notations that used in [11] to describe the access tree.

Access tree $\mathcal{T}$ Let $\mathcal{T}$ be an access tree, if num_x is the number of children of node x and k_x is its threshold value, then $0 < k_x \leq num_x$. Specially, when $k_x = 1$, the threshold gate is an OR gate and when $k_x = num_x$, the threshold gate is an AND gate.

Several functions are defined for convenience. We denote the parent of node x by $parent(x)$, and attribute value of node x by $att(x)$ if and only if x is a leaf node. Furthermore, the access tree $\mathcal{T}$ also defines the order between children of every node and uses function $index(x)$ to denote the number associated with node x from 1 to num_x.

Satisfying an access tree Let r be the root of access tree $\mathcal{T}$ and $\mathcal{T}_x$ be the subtree of $\mathcal{T}$. If a set of attributes S satisfy the access tree $\mathcal{T}_x$, we denote $\mathcal{T}_x(S) = 1$. $\mathcal{T}_x(S)$ is recursively computed as follows. If x is a interior node, compute $\mathcal{T}_{x'}(S)$ for all children x' of node x. $\mathcal{T}_x(S) = 1$ only when at least k_x children return 1. For leaf node x, $\mathcal{T}_x(S) = 1$ if and only if $att(x) \in S$.

Outline of N-authority CP-ABE An N-authority CP-ABE scheme consists of the following five algorithms:

Global setup$(1^\lambda) \rightarrow$ params. This algorithm takes as input an implicit security parameter λ and returns the system parameters params.

Authority setup$(1^\lambda) \rightarrow (SK_k, PK_k)$. This algorithm runs by authorities. Each authority A_k generates his secret key SK_k and public key PK_k for $k = 1, 2, \ldots, N$, namely there are totally N authorities in the system.

KeyGen$(SK_k, GID, S) \rightarrow SK_U$. This algorithm runs by each authority A_k with a user U to generate user's secret key. This algorithm takes as input the secret key SK_k of authority A_k, a set of attributes S and user's global identifier GID, and outputs user's secret key SK_U.

Encryption$(PK_k, M, \mathcal{T}) \rightarrow CT$. This algorithm takes as input a message M, public key PK_k of each authority A_k, and an access tree $\mathcal{T}$, returns the ciphertext CT. A user can decrypt the ciphertext CT if and only if the attribute set associated with secret key satisfies the access tree $\mathcal{T}$ associated with the ciphertext.

Decryption$(PK_k, SK_U, CT) \rightarrow M$. This algorithm takes as input public key PK_k of each authority A_k, user's secret key SK_U, and the ciphertext CT, if the set of attributes in user's secret key satisfies the access tree associated with ciphertext, this algorithm returns the message M.

Definition 2 We say that an N-Authority CP-ABE scheme is correct if

$$\Pr\left[\begin{array}{l|l} & \mathsf{Global\ Setup}(1^\lambda) \\ & \rightarrow params; \\ & \mathsf{Authorities\ Setup}(1^\lambda) \\ \mathsf{Decryption}(PK_k, & \rightarrow (SK_k, PK_k); \\ SK_U, CT) & \mathsf{KeyGen}(SK_k, GID, S) \\ = M & \rightarrow SK_U; \\ & \mathsf{Encryption}(PK_k, M, \mathcal{T}) \\ & \rightarrow CT; \\ & \mathcal{T}_r(S) = 1, \end{array}\right] = 1$$

where the probability is taken over the random coins of all the algorithms in the protocol.

Security model of N-authority CP-ABE The security model for N-authority CP-ABE is defined as follows:

Initialization Adversary $\mathcal{A}$ submits the access structure $\mathbb{A}^*$ he wishes to be challenged upon and a list of corrupted authorities $C_\mathcal{A}$ to algorithm $\mathcal{B}$.

Global setup The challenger runs the algorithm **Global Setup** and returns the system parameters params to the adversary $\mathcal{A}$.

Authorities setup For the corrupted authorities, the challenger sends his public and secret key (PK_k, SK_k) to the adversary $\mathcal{A}$. For the honest authorities, the challenger sends his public key PK_k to the adversary $\mathcal{A}$.

Phase 1 The adversary $\mathcal{A}$ queries for secret keys corresponding to attribute sets $S_1, S_2, \ldots, S_q$ where none of these keys satisfy the access structure $\mathbb{A}^*$ that adversary $\mathcal{A}$ selected.

Challenge The adversary $\mathcal{A}$ submits two equal-length messages M_0 and M_1. The challenger flips a random coin $b \in_R \{0, 1\}$ and runs the algorithm **Encryption** to get ciphertext CT^*. The challenger returns the ciphertext CT^* to adversary $\mathcal{A}$.

Phase 2 Phase 1 is repeated adaptively.

Guess Adversary $\mathcal{A}$ outputs his guess b' on b.

The advantage of an adversary $\mathcal{A}$ in this game is defined as $Pr[b' = b] - \frac{1}{2}$.

Definition 3 An N-authority CP-ABE scheme is (t, q, ϵ) secure in the above security model if no probabilistic polynomial-time adversary $\mathcal{A}$ making q secret key queries has advantage at least

$$\mathtt{Adv}_{\mathcal{A}}^{N-CP-ABE}(1^\lambda) = \left| Pr[b = b'] - \frac{1}{2} \right| > \epsilon(\lambda)$$

in the above security model.

2.2 Bilinear maps

Let $\mathbb{G}, \mathbb{G}_T$ be two multiplicative cyclic groups of prime order p. Let g be a generator of $\mathbb{G}$. In addition, let $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ denote the bilinear map. The bilinear map e has the following properties

- Bilinearity: for all $g, h \in \mathbb{G}, a, b \in \mathbb{Z}_p$, we have $e(g^a, h^b) = e(g, h)^{ab}$.
- Non-degeneracy: $e(g, g) \neq 1 \in \mathbb{G}_T$.
- Computability: Group operation $e(g, h)$ is efficiently computable, where $g, h \in \mathbb{G}$.

2.3 The decisional bilinear Diffie–Hellman (DBDH) assumption

Let g be a generator of $\mathbb{G}$. a, b, c, z are chosen at random in $\mathbb{Z}_p$. The DBDH assumption [6,29] is that no probabilistic polynomial-time algorithm $\mathcal{B}$ can distinguish the tuple $(g, A = g^a, B = g^b, C = g^c, e(g, g)^{abc})$ from the tuple $(g, A = g^a, B = g^b, C = g^c, e(g, g)^z)$ with more than a non-negligible advantage. The advantage of algorithm $\mathcal{B}$ is

$$\begin{aligned}\mathtt{Adv}_{\mathcal{B}}^{DBDH} &= |Pr[\mathcal{B}(A, B, C, g^{abc}) = 1] \\ &\quad - Pr[\mathcal{B}(A, B, C, g^z) = 1]|.\end{aligned}$$

3 Anonymous key issuing protocol

We use the same anonymous key issuing protocol introduced in [16] with the only difference that we add an extra component into the key. Let $u \in \mathbb{Z}_p$ be a private value of user and α, β, γ be private keys of authority. g, g_1, h are the generators of group $\mathbb{G}$. By running the anonymous ABE key issuing protocol, user U can finally work out his secret key $D = (g^\alpha h^r g_1^{1/(u+\beta)})^\gamma$.

Figure 1 shows the detailed anonymous ABE key issuing protocol. By $x \in_R X$, we denote that value x is randomly selected from X. The symbol PoK represents a proof of

Fig. 1 Our anonymous ABE key issuing protocol

User		Attribute Authority
$\rho_1 \in_R \mathbb{Z}_p$	$\xleftrightarrow{2PC}$	$x = (\beta + u)\rho_1, \tau, r \in_R \mathbb{Z}_p$
$\rho_2 \in_R \mathbb{Z}_p$	$\xleftarrow{X_1, X_2, X_3, PoK(\alpha,\tau,x,r)}$	$X_1 = g_1^{\tau/x}, X_2 = g^{\alpha\tau}, X_3 = h^{r\tau}$
$Y = (X_1^{\rho_1} X_2 X_3)^{\rho_2}$	$\xrightarrow{Y, PoK(\rho_2)}$	
$D = Z^{1/\rho_2}$	$\xleftarrow{Z, PoK(\tau,\gamma)}$	$Z = Y^{\gamma/\tau}$

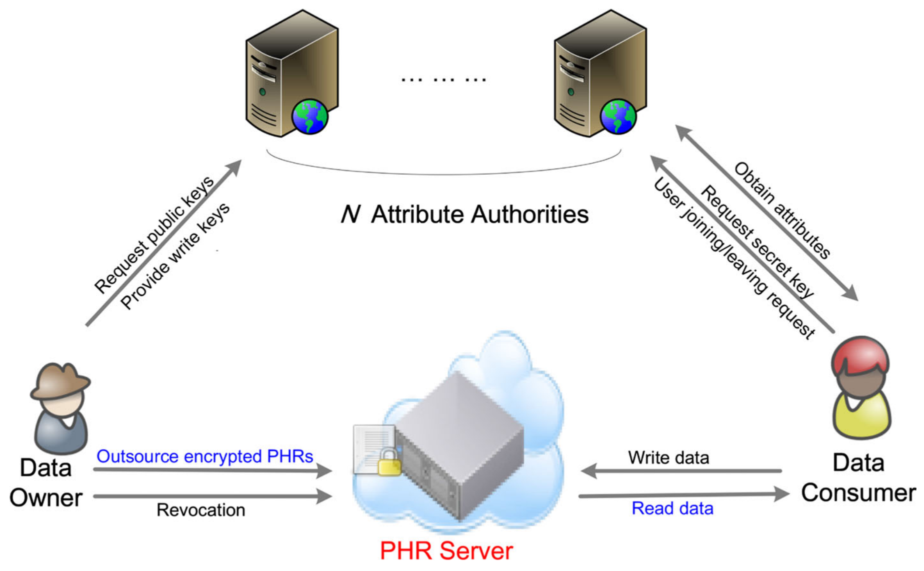

Fig. 2 Our PHR system model

knowledge of involved secret values. In the first step, by running a 2-party secure computation (2PC) protocol with inputs (u, ρ_1) from user and β from authority, authority can compute $x = (\beta + u)\rho_1 \bmod p$.

4 Our construction

In this section, we first give our PHR system model and then introduce the detailed description of our scheme. Our PHR system model is based on the system model proposed in [4], and we mainly focus on situations in PUDs. Our proposed scheme is a multi-authority CP-ABE scheme and allows on-demand user/attribute revocation.

4.1 Our PHR system model

In our PHR system, there are four essential parties *N Attribute Authorities*, *PHR Server*, *Data Owner*, and *Data Consumer*. An attribute authority is usually an organization that is responsible for supervising PHR information exchange. It could be a regional health information organization (RHIO) or other national department. PHR server stores encrypted PHRs with multiple cryptographic schemes including KP-ABE and CP-ABE. A data owner is a patient who owns PHRs. The patient should be able to create, manage, control, and share his PHRs with a wide range of data consumers. A data consumer is the person who is allowed to access patient's PHRs. Since we mainly consider the PUDs, data consumers in our PHR system are the ones who access patient's PHRs for professional purposes such as medical doctors and researchers. The system is illustrated in Fig. 2.

4.2 Scheme description

In the following part of this section, we will provide our detailed construction. We first give our N authority CP-ABE scheme and then add algorithms that can realize user/attribute revocation. Let λ be the security parameter that determines the size of groups used in our construction. Define the Lagrange coefficient $\Delta_{i,S}(x)$ for $i \in \mathbb{Z}_p$ and a set, S, of elements in $\mathbb{Z}_p$: $\Delta_{i,S}(x) = \prod_{j\in S, j\neq i} \frac{x-j}{i-j}$. For each attribute, associate a unique element in $\mathbb{Z}_p^*$ with that attribute. Algorithms of our N authority CP-ABE scheme are as follows:

Global setup Given the security parameter λ, this algorithm outputs an admissible bilinear group parameters $(e, p, g, h, h_1, \mathbb{G}, \mathbb{G}_T)$, where $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. $\mathbb{G}$, $\mathbb{G}_T$ are two multiplicative cyclic groups with prime order p, and g, h, h_1 are the generators of group $\mathbb{G}$. Suppose there are totally N authorities in the system, namely $A_1, A_2, \ldots, A_N$. Each authority A_k monitors a set of attributes $\tilde{A}_k = \{a_{k,1}, a_{k,2}, \ldots, a_{k,n_k}\}$. Then, this algorithm chooses a collusion-resistant hash function (CRHF) $H : \{0,1\}^* \rightarrow \mathbb{Z}_p$ and computes $u = H(GID)$ for each global identities GID.

Authority setup Each authority A_k chooses $\alpha_k \in_R \mathbb{Z}_p$ and computes $Y_k = e(g,g)^{\alpha_k}$. For each attribute $a_{k,i} \in \tilde{A}_k$, authority A_k chooses $t_{k,i} \in_R \mathbb{Z}_p^*$ and computes $T_{k,i} = g^{t_{k,i}}$. To generate PRF seeds, authorities A_k and A_j share a seed $s_{kj} \in_R \mathbb{Z}_p$ through a 2-party key exchange and keep the seed s_{kj} as a secret between them. Obviously, we have $s_{kj} = s_{jk}$. Each authority A_k chooses $x_k \in_R \mathbb{Z}_p$, computes $y_k = h_1^{x_k}$ and defines a pseudorandom function $PRF_{kj}(\cdot)$ that can only be computed by authority A_k and A_j as $PRF_{kj}(u) = h_1^{x_k x_j/(s_{kj}+u)}$. Each authority A_k initializes a version number $ver = 1$. Here, the version number will be used to realize revocation. Private key of authority A_k is $SK_k = \langle ver, \alpha_k, x_k, \{s_{kj}\}_{j\in\{1,\dots,N\}\backslash\{k\}}, \{t_{k,i}\}_{i\in\{1,\dots,n_k\}}\rangle$ and the public key of authority A_k is $PK_k = \langle ver, Y_k, y_k, \{T_{k,i}\}_{i\in\{1,\dots,n_k\}}\rangle$.

KeyGen Suppose that a user U with corresponding GID information u possesses a set of attributes $\tilde{A}_U$. Each authority A_k chooses $r_k \in_R \mathbb{Z}_p$ and computes $S_{k,i} = h^{r_k/t_{k,i}}$ for $a_{k,i} \in \tilde{A}_U^k$, where $\tilde{A}_U^k = \tilde{A}_U \cap \tilde{A}_k$. User U interacts with each authority A_k and calls the anonymous key issuing protocol introduced in Sect. 3 $N-1$ times for $\gamma = \delta_{kj}, \alpha = \delta_{kj}\alpha_k, r = \delta_{kj}r_k, g_1 = y_j^{x_k}$ and $\beta = s_{kj}$, where $j \in \{1,\dots,N\}\backslash\{k\}$. If $k > j$, we denote $\delta_{kj} = 1$, otherwise $\delta_{kj} = -1$. Therefore, we have $D_{kj} = g^{\alpha_k}h^{r_k}PRF_{kj}(u)$ for $k > j$ and $D_{kj} = g^{\alpha_k}h^{r_k}/PRF_{kj}(u)$ for $k < j$. Finally, user U computes $D_U = \prod_{(k,j)\in\{1,\dots,N\}\times(\{1,\dots,N\}\backslash\{k\})} D_{kj} = g^{\sum_{k\in[1,\dots,N]}(N-1)\alpha_k}h^{\sum_{k\in\{1,\dots,N\}}(N-1)r_k}$. Therefore, the secret key of user U is

$$SK_U = \langle ver, D_U, \{S_{k,i}\}_{k\in\{1,\dots,N\},a_{k,i}\in\tilde{A}_k^U}\rangle.$$

Encryption To encrypt message M under the specified tree access structure $\mathcal{T}$, this algorithm first determines a polynomial q_x for each node x in the tree $\mathcal{T}$ and sets the polynomial q_x with degree $k_x - 1$ for each node x, where k_x is the threshold value of that node. Starting from the root node r, this algorithm chooses $s \in_R \mathbb{Z}_p$ and sets $q_r(0) = s$. Then, it chooses other points and completes the polynomial q_r. For any other node x, this algorithm sets $q_x(0) = q_{parent(x)}(index(x))$ and randomly chooses other points to complete the polynomial q_x. Let $\tilde{A}_{\mathcal{T}}$ be the set of leaf nodes in tree access structure $\mathcal{T}$, the ciphertext is constructed as

$$CT = \left\langle ver, C_1 = M\left(\prod_{k\in\{1,\dots,N\}} Y_k\right)^s, \right.$$
$$\left. C_2 = g^s, \left\{C_{k,x} = T_{k,i}^{q_x(0)}\right\}_{a_{k,i}\in\tilde{A}_{\mathcal{T}}} \right\rangle,$$

where $a_{k,i} = att(x)$ is the attribute of leaf node x. The encryptor stores the number s that is randomly selected by himself.

Decryption In order to decrypt the ciphertext CT, we first define a recursive algorithm $DecryptNode(CT, SK_U, x)$ that takes the ciphertext $CT = \langle ver, C_1, C_2, \{C_{k,i}\}_{a_{k,i}\in\tilde{A}_{\mathcal{T}}}\rangle$, user's secret key $SK_U = \langle ver, D_U, \{S_{k,i}\}_{k\in\{1,\dots,N\},a_{k,i}\in\tilde{A}_k^U}\rangle$, and a node x from $\mathcal{T}$ as input. If $a_{k,i} \in \tilde{A}_k^U$, then

$$\begin{aligned} DecryptNode(CT, SK_U, x) &= \prod_{k\in\{1,\dots,N\}} e(C_{k,x}, S_{k,i}) \\ &= \prod_{k\in\{1,\dots,N\}} e(T_{k,i}^{q_x(0)}, h^{r_k/t_{k,i}}) \\ &= e(g,h)^{(\sum_{k\in\{1,\dots,N\}} r_k)\cdot q_x(0)}. \end{aligned}$$

If $a_{k,i} \notin \tilde{A}_k^U$, we define $DecryptNode(CT, SK_U, x) = \perp$.

If node x is not a leaf node, then the algorithm $DecryptNode(CT, SK_U, x)$ proceeds recursively as follows: For all children nodes z of node x, it calls $DecryptNode(CT, SK_U, z)$ and stores the output as F_z. Let S_x be an arbitrary k_x-sized set of child nodes z such that $F_z \neq \phi$. If no such set exists, then we say the node was not satisfied and returns $\perp$. Otherwise, we compute

$$\begin{aligned} F_x &= \prod_{z\in S_x} F_z^{\Delta_{d,S_x'}(0)}, \text{ where } \begin{array}{l} d = \text{index}(z) \\ S_x' = \{\text{index}(z) : z \in S_x\} \end{array} \\ &= \prod_{z\in S_x} (e(g,h)^{(\sum_{k\in\{1,\dots,N\}} r_k)\cdot q_z(0)})^{\Delta_{d,S_x'}(0)} \\ &= \prod_{z\in S_x} (e(g,h)^{(\sum_{k\in\{1,\dots,N\}} r_k)\cdot q_{parent(z)}(\text{index}(z))})^{\Delta_{d,S_x'}(0)} \\ &= \prod_{z\in S_x} e(g,h)^{(\sum_{k\in\{1,\dots,N\}} r_k)\cdot q_x(d)\Delta_{d,S_x'}(0)} \\ &= e(g,h)^{(\sum_{k\in\{1,\dots,N\}} r_k)\cdot q_x(0)}. \end{aligned}$$

If the tree is satisfied, we observe

$$DecryptNode(CT, SK_U, r) = e(g,h)^{(\sum_{k\in\{1,\dots,N\}} r_k)\cdot s}.$$

Then, user U computes

$$\begin{aligned} E &= e(D_U, C_2) \\ &= e(g^{\sum_{k\in\{1,\dots,N\}}(N-1)\alpha_k}h^{\sum_{k\in\{1,\dots,N\}}(N-1)r_k}, g^s) \\ &= e(g,g)^{s\sum_{k\in\{1,\dots,N\}}(N-1)\alpha_k}e(g,h)^{s\sum_{k\in\{1,\dots,N\}}(N-1)r_k}, \end{aligned}$$

Therefore, we have

$$\begin{aligned} & C_1 \cdot \frac{e(g,h)^{(\sum_{k\in\{1,\dots,N\}} r_k)\cdot s}}{E^{1/(N-1)}} \\ &= \frac{M\cdot(\prod_{k\in\{1,\dots,N\}} Y_k)^s e(g,h)^{(\sum_{k\in\{1,\dots,N\}} r_k)\cdot s}}{e(g,g)^{\sum_{k\in\{1,\dots,N\}} s\alpha_k}e(g,h)^{\sum_{k\in\{1,\dots,N\}} sr_k}} \end{aligned}$$

$$= \frac{Me(g,g)^{\sum_{k\in\{1,\dots,N\}} s\alpha_k} e(g,h)^{\sum_{k\in\{1,\dots,N\}} sr_k}}{e(g,g)^{\sum_{k\in\{1,\dots,N\}} s\alpha_k} e(g,h)^{\sum_{k\in\{1,\dots,N\}} sr_k}}$$
$$= M.$$

To realize user/attribute revocation and policy update, the following algorithms **ReKeyGen**, **ReEnc**, **KeyUpdate, and PolicyUpdate** will be used. Details of these algorithms are as follows:

ReKeyGen Given a set of attributes L that are needed to be updated and system parameters. For each $a_{k,i} \in L \cap \tilde{A}_k$, authority A_k chooses $t'_{k,i} \in_R \mathbb{Z}_p$, computes $T'_{k,i} = g^{t'_{k,i}}$ and $rk_{i\leftrightarrow i'} = \frac{t'_{k,i}}{t_{k,i}}$. Authority A_k outputs local re-key as $rk_k = (ver, \{rk_{k,i\leftrightarrow i'}\}_{a_{k,i}\in\tilde{A}_k})$. The global re-key is $rk = \{rk_k\}_{1\le k\le N}$. Increase the system's ver by 1.

ReEnc This algorithm is executed by the server and takes as input the original ciphertext CT, the global re-key rk, and a set of attributes W, which includes all the attributes in the original access structure $\mathcal{T}$. The server first checks the version number in ciphertext CT and re-key rk. If it is the latest version, server directly outputs CT. Otherwise, for each $a_{k,i} \in W$, the server recomputes the ciphertext. Assume the latest updated value corresponding to $a_{k,i}$ is $t_{k,i^{(n)}}$, then we can use $rk_{i\leftrightarrow i^{(n)}} = rk_{i\leftrightarrow i'} \cdot rk_{i'\leftrightarrow i''} \cdots rk_{i^{(n-1)}\leftrightarrow i^{(n)}} = \frac{t_{k,i^{(n)}}}{t_{k,i}}$ to compute $C^{(n)}_{k,x} = (C_{k,x})^{rk_{i\leftrightarrow i^{(n)}}} = g^{rkt_{k,i^{(n)}}}$. Ciphertext is output as $CT' = \langle ver+1, C_1, C_2, \{C'_{k,x}\}_{a_{k,i}\in\tilde{A}_{\mathcal{T}}}\rangle$, where $C'_{k,x} = C^{(n)}_{k,x}$ is the latest ciphertext.

KeyUpdate This algorithm takes as input the user's secret key SK^k_U, re-key rk, and the set of attributes W, which includes all the attributes in the original access structure $\mathcal{T}$. The server first checks the version number in user's secret key SK_U and re-key rk. If it is the latest version, server directly outputs SK_U. Otherwise, for each $a_{k,i} \in W$, the server computes $S^{(n)}_{k,i} = (S_{k,i})^{1/rk_{i\leftrightarrow i^{(n)}}} = h^{\frac{r_k}{t_{k,i^{(n)}}}}$. Secret key is output as $SK_U = \langle ver+1, D_U, \{S'_{k,i}\}_{k\in\{1,\dots,N\}, a_{k,i}\in\tilde{A}^U_k}\rangle$, where $S'_{k,i} = S^{(n)}_{k,i}$ is the latest secret key.

PolicyUpdate To reduce the computation costs, when updating policies, only the ciphertext corresponding to the added/changed attributes is updated. Namely, only for the added/changed attributes, computes $\tilde{C}_{k,x}$. Finally, this algorithm outputs $\{\tilde{C}T = \langle ver, C_1, C_2, \tilde{C}_{k,x}\}_{a_{k,i}\in\tilde{A}_{\mathcal{T}}}\rangle$.

Construction of secret keys $S_{k,i}$

Here, we claim that user cannot use secret keys $S_{k,i}$, which he received from authority A_k to construct a secret key that he should not possess. For example, we assume that authority A_1 monitors a set of attributes $\{a_{1,1}, a_{1,2}, a_{1,3}\}$, and a user possesses a set of attributes $\{a_{1,1}, a_{1,2}\}$. Authority A_1 chooses $t_{1,i} \in_R \mathbb{Z}^*_p$ for each attribute $a_{1,i}$, where $i \in \{1,2,3\}$. To generate the user's secret key, authority A_1 chooses $r_1 \in_R \mathbb{Z}_p$ and computes $S_{1,1} = h^{r_1/t_{1,1}}$, $S_{1,2} = h^{r_1/t_{1,2}}$. User should not be able to compute secret key $S_{1,3}$ for attribute $a_{1,3}$ as

$$S_{1,3} = \frac{S_{1,1}}{S_{1,2}} = \frac{h^{r_1/t_{1,1}}}{h^{r_1/t_{1,2}}}$$
$$= h^{\frac{r_1}{t_{1,1}t_{1,2}/(t_{1,2}-t_{1,1})}}$$
$$= h^{r_1/t_{1,3}}$$

or similar ways. This assumption holds with given probability where N_i is the number of situations that a valid secret key can be computed when user chooses i secret keys. The symbol C^k_n represents a combination, which is computed as $C^k_n = \frac{n!}{k!(n-k)!}$. Here, we consider the worst case when user possesses $n_k - 1$ attributes out of total n_k attributes that authority A_k monitors. $1 - \frac{1}{p-1}\left(\frac{N_2}{C^2_{p-1}} + \frac{N_3}{C^3_{p-1}} + \ldots + \frac{N_{n_k-1}}{C^{n_k-1}_{p-1}}\right) > 1 - \frac{1}{p-1} \cdot \frac{N_2\cdot(n_2-2)}{C^2_{p-1}} > 1 - \frac{n_k-2}{(p-1)^2}$.

Enforce write access control In the above scheme, we only grant read access control. However, authorized users may write something to patient's PHRs (for example, a doctor may write some comments on a patient's PHRs). A simple way to realize write access control is to encrypt the comments by using public key and send the ciphertext to the patient if there is no restriction on write access control. However, if the system allows anyone to write without verification on writer's identity, this may raise a lot of problems. An unauthorized writer may pretend as a doctor and give the comments. So this solution is undesirable. One way to overcome this problem is to use signature. Each time an authorized user wants to write, he obtains a signature from his organization to grant write access control. In [30], authors constructed two keys, signature key and read key, and encrypted these keys into ciphertext. Each time user want to write, a writing request is sent to server, then server will send the entire ciphertext including signature key, read key, and signature to the user. This solution requires organization or server always online. In [4], authors proposed a solution that do not need organization or server always online. They generate a hash chain and broadcast signature with hash value each time period. By deploying the techniques in [30] and [4] into our scheme, our scheme can also enforce write access control.

Enforce keywords search Our scheme is designed to be used in PUDs, where multi-authorities monitor sets of attributes and grant read/write access to authorized users like doctors and researchers. Take a heart disease researcher as example, if the system does not provide keywords search, then when the researcher wants to search cases of genetic heart disease, he has to first download and decrypt all the files and then search the files he needed. Therefore, it is desirable to grant keywords search in PUDs. Recently, multiple keywords search solutions have been proposed in [3,31,32]. By using the technique described in [3], our scheme can also enforce keywords search.

5 Security analysis and comparisons with existing scheme

In this section, we first analyze the security of our scheme and then compare our scheme with other multi-authority ABE.

5.1 Security analysis

Fine-grainedness of access control The access structure used in our scheme is access tree structure which is very expressive and flexible. Thus, the data owner can specify a detailed expressive access tree to decide what kind of users can access the encrypted data files.

Collusion resistance In our scheme, authority A_k and A_j generate a PRF seed s_{kj} and keep it as a secret between them. Even if there is up to $(N-2)$ corrupted authorities, two PRF seeds are still kept unknown to the adversary. We also add authority's private key into user secret key. When one of the last two authorities A_k and A_j is malicious, the adversary still cannot compute a valid secret key. Therefore, our scheme can resist up to $(N-1)$ corrupted authorities attacks. To address the user privacy problem in cloud computing environment, we introduce the anonymous key issuing protocol. By using anonymous key issuing protocol, corrupted authorities can get nothing about user's GID, and thus cannot collect user's attributes by tracing user's GID. Hence, the corrupted authorities cannot get user's privacy information.

Forward secrecy Forward secrecy means that a revoked user who lose an attribute or part of access structure will not be able to access the plaintext of the subsequent data exchanged. Suppose a user or attribute is revoked, then next time when user access the server, our scheme will encrypt any new files uploaded to the server after the revocation using updated public key and re-encrypt existing files using re-key. The secret keys corresponding to revoked components will not update, so revoked users will not be able to access plaintext using secret keys that is not updated. Therefore, our scheme achieves forward secrecy.

Data confidentiality We analyze data confidentiality of our scheme by comparing it with an original scheme that removes the revocation part of our scheme. We first prove the security of the original scheme as follows and define the semantic security game as *Game*0. Then, similar to [4,23], we will use a series of games to reduce our scheme to the original scheme.

Theorem 1 *Our N-authority CP-ABE scheme is $(t, q, \epsilon(\lambda))$ semantically secure in the security model described in Sect. 2, if the $(t', \epsilon'(\lambda))$ decisional bilinear Diffie–Hellman assumption holds in $(e, p, \mathbb{G}, \mathbb{G}_T)$, where*

$$t' = t + q\cdot\mathcal{O}(t) \text{ and } \epsilon'(\lambda) = \frac{\epsilon}{2}\prod_{k\in\{1,\ldots,N\}}\left(1-\frac{n_k-2}{(p-1)^2}\right).$$

Proof Suppose there exists a polynomial-time adversary $\mathcal{A}$ who can break our scheme in the security model described in Sect. 2 with advantage ϵ. We can construct a simulator $\mathcal{B}$ that will break the DBDH assumption with advantage $\frac{\epsilon}{2}\prod_{k\in\{1,\ldots,N\}}(1-\frac{n_k-2}{(p-1)^2})$, where n_k is the number of attributes that authority A_k monitors.

The challenger generates the admissible bilinear group parameters $(e, p, \mathbb{G}, \mathbb{G}_T)$ with prime order p and chooses the random generators $g, h, h_1 \in \mathbb{G}$. The challenger flips an unbiased binary coin μ. If $\mu = 0$, the challenger sends $(A, B, C, Z) = (g^a, g^b, g^c, g^{abc})$ to $\mathcal{B}$; otherwise, it sends $(A, B, C, Z) = (g^a, g^b, g^c, g^z)$ to $\mathcal{B}$, where $a, b, c, z \in_R \mathbb{Z}_p$.

Initialization Adversary $\mathcal{A}$ submits the access structure $\mathbb{A}^*$ he wishes to be challenged upon and a list of corrupted authorities $C_{\mathcal{A}}$ to algorithm $\mathcal{B}$, where $|C_{\mathcal{A}}| < N$. Algorithm $\mathcal{B}$ chooses $\eta \in_R \mathbb{Z}_p$ and sets $h = Ag^{\eta}$.

Authorities setup $\mathcal{B}$ randomly chooses $A_k^* \in \{A_1, A_2, \ldots, A_N\}\backslash C_{\mathcal{A}}$.

(1) For $A_k \in C_{\mathcal{A}}$, $\mathcal{B}$ chooses $v_k, w_{k,i} \in_R \mathbb{Z}_p$ and sets $T_{k,i} = g^{w_{k,i}}$ for $a_{k,i} \in \tilde{A}_k$. Then, $\mathcal{B}$ chooses a value $x_k \in_R \mathbb{Z}_p$, a PRF seed $s_{kj} \in_R \mathbb{Z}_p$ for corrupted authorities A_k and A_j, and computes $y_k = h_1^{x_k}$. $\mathcal{B}$ sends $< v_k, x_k, s_{k,j}, w_{k,i} >$ and $< Y_k, y_k, T_{k,i} >$ to adversary $\mathcal{A}$, where Y_k is computed as $Y_k = e(g, g)^{v_k}$.

(2) For $A_k \notin C_{\mathcal{A}}$, $\mathcal{B}$ chooses $v_k, w_{k,i} \in_R \mathbb{Z}_p$, sets $T_{k,i} = g^{w_{k,i}}$ for $a_{k,i} \in \mathbb{A}^*$ and $T_{k,i} = h^{w_{k,i}} = g^{(a+\eta)w_{k,i}}$ for $a_{k,i} \notin \mathbb{A}^*$. Then, $\mathcal{B}$ chooses a value $x_k \in_R \mathbb{Z}_p$ and computes $y_k = h_1^{x_k}$. If $A_k \neq A_k^*$, $\mathcal{B}$ sets $Y_k = e(g, g)^{bv_k}$; otherwise, $\mathcal{B}$ sets $Y_k = e(g, g)^{ab}\prod_{A_k\in C_{\mathcal{A}}} e(g, g)^{-v_k}$ $\prod_{A_k\notin C_{\mathcal{A}}, A_k\neq A_k^*} e(g, g)^{-bv_k}$. For two honest authority A_k and A_j, $\mathcal{B}$ chooses a PRF seed $s_{kj} \in_R \mathbb{Z}_p$ for them and sends $\langle Y_k, y_k, T_{k,i}\rangle$ to adversary $\mathcal{A}$.

Phase 1 The adversary $\mathcal{A}$ queries for secret keys corresponding to attribute sets $S_1, S_2, \ldots, S_q$ where none of these keys satisfy the access structure $\mathbb{A}^*$ that adversary $\mathcal{A}$ selected.

(1) $A_k \in C_{\mathcal{A}}$, $\mathcal{B}$ uses $\langle v_k, x_k, s_{k,j}, w_{k,i}\rangle$ to compute secret keys for corresponding attribute sets.

(2) For $A_k \notin C_{\mathcal{A}}$, $\mathcal{B}$ chooses $r_k \in_R \mathbb{Z}_p$, computes $\{S_{k,i} = h^{r_k/w_{k,i}}\}_{a_{k,i}\in\mathbb{A}^*}$ and $\{S_{k,i} = h^{r_k/((a+\eta)w_{k,i})}\}_{a_{k,i}\notin\mathbb{A}^*}$. Secret key D_{kj} is computed by $\mathcal{B}$ in two different situations as follows:

– $A_k \neq A_k^*$: For $k > j$, set $D_{kj} = B^{v_k} h^{r_k} PRF_{kj}(u')$; otherwise, set $D_{kj} = B^{v_k} h^{r_k} / PRF_{kj}(u')$.
– $A_k = A_k^*$: For $k > j$, set $D_{kj} = B^{-\eta} \prod_{A_k \in C_{\mathcal{A}}} g^{-v_k} \prod_{A_k \notin C_{\mathcal{A}}, A_k \neq A_k^*} B^{-v_k} h^{r_k} PRF_{kj}(u')$; otherwise, set $D_{kj} = B^{-\eta} \prod_{A_k \in C_{\mathcal{A}}} g^{-v_k} \prod_{A_k \notin C_{\mathcal{A}}, A_k \neq A_k^*} B^{-v_k} h^{r_k} / PRF_{kj}(u')$.

We claim that D_{kj} is correctly distributed. Here, we only describe the situation when $k > j$ since the situation $k < j$ is as simple as the situation when $k > j$.

$$\begin{aligned} D_{kj} &= B^{-\eta} \prod_{A_k \in C_{\mathcal{A}}} g^{-v_k} \prod_{A_k \notin C_{\mathcal{A}}, A_k \neq A_k^*} B^{-v_k} h^{r_k} PRF_{kj}(u') \\ &= g^{-b\eta} (g^a g^\eta)^{r_k} g^{-(\sum_{A_k \in C_{\mathcal{A}}} v_k + \sum_{A_k \notin C_{\mathcal{A}}, A_k \neq A_k^*} bv_k)} \\ &\quad \times PRF_{kj}(u') \\ &= (g^a g^\eta)^{-b} g^{ab} g^{-(\sum_{A_k \in C_{\mathcal{A}}} v_k + \sum_{A_k \notin C_{\mathcal{A}}, A_k \neq A_k^*} bv_k)} \\ &\quad \times (g^a g^\eta)^{r_k} PRF_{kj}(u') \\ &= g^{ab} (g^a g^\eta)^{r_k - b} g^{-(\sum_{A_k \in C_{\mathcal{A}}} v_k + \sum_{A_k \notin C_{\mathcal{A}}, A_k \neq A_k^*} bv_k)} \\ &\quad \times PRF_{kj}(u') \\ &= g^{ab - (\sum_{A_k \in C_{\mathcal{A}}} v_k + \sum_{A_k \notin C_{\mathcal{A}}, A_k \neq A_k^*} bv_k)} h^{r_k - b} \\ &\quad \times PRF_{kj}(u'). \end{aligned}$$

Let $r'_k = r_k - b$, we have

$$D_{kj} = g^{ab - (\sum_{A_k \in C_{\mathcal{A}}} v_k + \sum_{A_k \notin C_{\mathcal{A}}, A_k \neq A_k^*} bv_k)} h^{r'_k} PRF_{kj}(u').$$

Challenge Adversary $\mathcal{A}$ submits two equal-length messages M_0 and M_1 to the simulator $\mathcal{B}$. $\mathcal{B}$ flips a random coin $\nu \in_R \{0, 1\}$ and computes

$$C_1^* = M_\nu \cdot Z, C_2^* = C, \{C_{k,x}^* = T_{k,i}^{q_x(0)}\}_{a_{k,i} \in \mathbb{A}^*}.$$

$\mathcal{B}$ sends the ciphertext $C_{T,\nu}^* = (C_1^*, C_2^*, \{C_{k,x}^*\}_{a_{k,i} \in \mathbb{A}^*})$ to the adversary $\mathcal{A}$.

If $\mu = 0$, then $Z = g^{abc}$. If we let $s = c$, then we can compute

$$\begin{aligned} \prod_{k \in \{1,\ldots,N\}} {Y_k}^c &= \prod_{A_k \in C_{\mathcal{A}}} e(g,g)^{cv_k} \prod_{A_k \notin C_{\mathcal{A}}, A_k \neq A_k^*} e(g,g)^{cbv_k} \\ & (e(g,g)^{abc} \prod_{A_k \in C_{\mathcal{A}}} e(g,g)^{-cv_k} \prod_{A_k \notin C_{\mathcal{A}}, A_k \neq A_k^*} e(g,g)^{-cbv_k}) \\ &= e(g,g)^{abc} \\ &= Z. \end{aligned}$$

So $C_{T,\nu}^*$ is a valid ciphertext of message M_ν.

Phase 2 Phase 1 is repeated adaptively.

Guess The adversary $\mathcal{A}$ outputs his guess ν' on ν. If $\nu' = \nu$, $\mathcal{B}$ outputs his guess $\mu' = 0$ on μ. Otherwise, $\mathcal{B}$ outputs $\mu' = 1$ on μ.

Since the input of Z is a random number z when $\mu = 1$, the adversary $\mathcal{A}$ gains no information about ν. Therefore, adversary $\mathcal{A}$ can distinguish ν with no advantage, namely $Pr[\nu' \neq \nu | \mu = 1] = \frac{1}{2}$. $\mathcal{B}$ outputs his guess $\mu' = 1$ when $\nu' \neq \nu$, thus we have $Pr[\mu' = \mu | \mu = 1] = \frac{1}{2}$.

If $\mu = 0$, then the advantage of adversary $\mathcal{A}$ outputs $\nu' = \nu$ is at least ϵ by definition. Therefore, we have $Pr[\nu' = \nu | \mu = 0] \geq \frac{1}{2} + \epsilon$. When $\nu' = \nu$, $\mathcal{B}$ outputs his guess $\mu' = 0$ on μ, so we have $Pr[\mu' = \mu | \mu = 0] \geq \frac{1}{2} + \epsilon$.

Therefore, algorithm $\mathcal{B}$'s advantage to break DBDH assumption is $\prod_{k \in \{1,\ldots,N\}} (1 - \frac{n_k - 2}{(p-1)^2}) | \frac{1}{2} Pr[\mu' = \mu | \mu = 0] + \frac{1}{2} Pr[\mu' = \mu | \mu = 1] - \frac{1}{2} | \geq \frac{\epsilon}{2} \prod_{k \in \{1,\ldots,N\}} (1 - \frac{n_k - 2}{(p-1)^2})$. □

Game 1 This game is different from $Game0$, more than one (PK_k, MK_k) versions are defined in this game. Given the $PK_k s$, the re-key between any two versions, and user secret keys that does not satisfy the challenge access structure with a set of attributes submitted by the adversary $\mathcal{A}$.

Game 2 This game is the security game for our proposed scheme. The only difference between $Game2$ and $Game1$ is that only partial user secret keys are given to the adversary $\mathcal{A}$ in this game.

Lemma 1 *The advantage of the adversary in $Game1$ is the same as that in $Game0$.*

Proof If there exists a polynomial-time algorithm $\mathcal{A}$ can win the $Game1$ with non-negligible advantage, then we can construct a polynomial-time algorithm $\mathcal{B}$ to win the $Game0$ with non-negligible advantage. The main idea is described as follows. In the security model, adversary submits two equal-length message M_0 and M_1. Challenger flips a random coin $b \in_R \{0, 1\}$, computes the ciphertext of M_b, and sends the challenge ciphertext to adversary. Adversary wins the game if he outputs his guess b' on b with $b' = b$. The only difference between $Game0$ and $Game1$ is that challenger gets secret keys for more versions in $Game1$. If adversary can win the $Game1$ and $\mathcal{B}$ outputs $b' = b$, then $\mathcal{A}$ also wins $Game0$, namely $Adv_{\mathcal{B}}(Game1) \leq Adv_{\mathcal{A}}(Game0)$. On the other hand, adversary gets more information in $Game1$ than in $Game0$; thus, we have $Adv_{\mathcal{A}}(Game0) \leq Adv_{\mathcal{B}}(Game1)$. Therefore, we have $Adv_{\mathcal{B}}(Game1) = Adv_{\mathcal{A}}(Game0)$. □

Lemma 2 *Given the authority A_k^* that is not corrupted, the advantage of the adversary in $Game2$ is the same as that in $Game1$.*

Proof The only difference between $Game2$ and $Game1$ is that partial user secret key is given to adversary in $Game2$. However, these partial secret keys are equivalent to the secret keys queried in $Game1$, namely all the secret queries made by adversary in $Game1$ can be made in $Game2$. Therefore, from views of the adversary, $Game2$ and $Game1$ are the same. □

Table 1 Comparisons of existing scheme

Scheme	KP/CP-ABE	Security	Central authority	Access policy	Revocation means	Privacy-preserving
Chase [14]	KP-ABE	Not against user–server collusion	Yes	Threshold	N/A	No
Lin et al. [15]	KP-ABE	Not against user–server collusion	No	Threshold	N/A	No
Chase et al. [16]	KP-ABE	Against N-2 authority collusion	No	Threshold	N/A	Yes
Our scheme	CP-ABE	Against N-1 authority collusion	No	Access tree	Attribute-level, immediate	Yes

5.2 Comparisons with existing scheme

We compare our scheme with other multi-authority ABE [14–16] in Table 1. From the table, we can see that our scheme is a multi-authority CP-ABE without central authority. Our scheme achieves high privacy protection, while up to $N-1$ authorities are corrupted, our scheme remains secure. Moreover, our scheme allows on-demand user/attribute revocation. The access policy in our scheme is expressive access tree structure, which therefore guarantee fine-grained access control. Compared with scheme [14], our scheme removes the central authority, which is responsible for issuing public and private keys to other authorities and thus able to decrypt all the ciphertexts in the system. Compared with scheme [15], our scheme can protect user's privacy. In [15], when corrupted authorities work together, they can collect user's attributes by tracing his GID, thus get user's privacy. In our scheme, by using an anonymous key issuing protocol, we generate user's secret key without leaking anything about user's GID to authorities. Compared with [16], our scheme allows immediate user/attribute revocation and our scheme is secure against $N-1$ authority collusion.

6 Conclusions

In this paper, we propose a privacy-preserving multi-authority CP-ABE scheme that can be used in PUDs of PHR system [4] in cloud computing. When encrypting PHRs, patient can associate an expressive access tree structure with the ciphertext, thus achieving fine-grained access control. We also achieve privacy-preserving by using anonymous key issuing protocol. Corrupted authorities can get nothing about user's GID while executing anonymous key issuing protocol, and therefore, they cannot collect user's attributes by tracing GID. Furthermore, our scheme supports efficient and on-demand lazy user revocation, which reduce the overhead a lot. We prove the security of our scheme under a standard complexity assumption (respectively, DBDH).

Acknowledgments We would like to thank anonymous referees for their helpful comments and suggestions. This work is supported by the National Natural Science Foundation of China (61272542, 61300213), the Fundamental Research Funds for the Central Universities (2013B07014).

References

1. Fernandes, Diogo A.B., Soares, Liliana F.B., et al.: Security issues in cloud environments: a survey. Int. J. Inf. Secur. **13**(2), 113–170 (2014)
2. Gouglidis, A., Mavridis, I., Hu, V.C.: Security policy verification for multi-domains in cloud systems. Int. J. Inf. Secur. **13**(2), 97–111 (2014)
3. Li, M., Yu, S., Cao, N., Lou, W.: Authorized private keyword search over encrypted personal health records in cloud computing. In: Proceedings of the 31st IEEE International Conference on Distributed Computing Systems (ICDCS'11), pp. 383–392 (2011)
4. Li, M., Yu, S., Zheng, Y., Ren, K., Lou, W.: Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption. IEEE Trans. Parallel Distrib. Syst. **24**, 131–143 (2013)
5. Health insurance portability and accountability act of 1996. U.S. Government Printing Office (1996)
6. Sahai, A., Waters, B.: Fuzzy identity based encryption. In: Advances in Cryptology—EUROCRYPT 2005, LNCS 3494, pp. 457–473 (2005)
7. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: Proceedings of IEEE Symposium on Security and Privacy 2007 (SP'07), LNCS 6571, pp. 321–334 (2007)
8. Cheung, L., Newport, C.: Provably secure ciphertext policy ABE. In: Proceedings of the 14th ACM Conference on Computer and Communications Security (CCS'07) pp. 456–465 (2007)
9. Herranz, J., Laguillaumie, F., Ràfols, C.: Constant size ciphertexts in threshold attribute-based encryption. In: Proceedings of 13th International Conference on Practice and Theory in Public Key Cryptography (PKC'10) pp. 19–34 (2010)
10. Waters, B.: Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In: Proceedings of 14th International Conference on Practice and Theory in Public Key Cryptography (PKC'11), LNCS, Vol. 6571, pp. 53–70. Springer-Verlag, Berlin Heidelberg New York (2011)
11. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted Data. In: Proceedings of the 13th ACM Conference on Computer and Communications Security (CCS'06) x, pp. 89–98 (2006)
12. Ostrovsky, R., Sahai, A., Waters, B.: Attribute-based encryption with non-monotonic access structures. In: Proceedings of the 14th ACM Conference on Computer and Communications Security (CCS'07) pp. 195–203 (2007)
13. Mandl, K.D., Szolovits, P., Kohane, I.S.: Public standards and patients control: how to keep electronic medical records accessible but private. BMJ **322**(7281), 283–287 (2001)
14. Chase, M.: Multi-authority attribute based encryption. In: Proceedings of the 4th Theory of Cryptography Conference (TCC'07) pp. 515–534 (2007)
15. Lin, H., Cao, Z., Liang, X., Shao, J.: Secure threshold multi-authority attribute based encryption without a central authority. In: Proceedings of the 9th International Conference on Cryptology in India (INDOCRYPT'08), pp. 426–436. (2008)

16. Chase, M., Chow, S.S.M.: 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 (2009)
17. Pirretti, M., Traynor, P., McDaniel, P., Waters, B.: Secure attribute-based systems. In: Proceedings of the 13th ACM Conference on Computer and Communications Security (CCS'06) pp. 99–112 (2006)
18. Boldyreva, A., Goyal, V., Kumar, V.: Identity-based encryption with efficient revocation. In: Proceedings of the 15th ACM Conference on Computer and Communications Security (CCS'05) pp. 417–426 (2008)
19. Liang, X., Lu, R., Lin, X., Shen, X.S.: Ciphertext Policy Attribute Based Encryption with Efficient Revocation. Univ. of Waterloo, Technical report (2010)
20. Yu, S., Wang, C., Ren, K., Lou, W.: Attribute based data sharing with attribute revocation. In: Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security (ASIACCS'10) pp. 261–270 (2010)
21. Hur, J., Noh, D.K.: Attribute-based access control with efficient revocation in data outsourcing system. IEEE Trans. Parallel Distrib. Syst. **22**, 1214–1221 (2011)
22. Jahid, S., Mittal, P., Borisov, N.: Easier: encryption-based access control in social networks with efficient revocation. In: Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security (ASIACCS'11) pp. 411–415 (2011)
23. Yu, S., Wang, C., Ren, K., Lou, W.: Achieving secure, scalable, and fine-grained data access control in cloud computing. In: Proceedings of the 29th IEEE International Conference on Computer Communications (INFOCOM'10) pp. 534–542 (2010)
24. Ibraimi, L., Petkovic, M., Nikova, S., Hartel, P., Jonker, W.: Ciphertext-Policy Attribute-Based Threshold decryption with Flexible Delegation and Revocation of User Attributes. University of Twente, Technical report (2009)
25. Ibraimi, L., Asim, M., Petkovic, M.: Secure Management of Personal Health Records by Applying Attribute-Based Encryption. University of Twente, Technical report (2009)
26. Akinyele, A., Lehmann, C.U., Green, M.D., Pagano, M.W., Peterson, Z.N.J., Rubin, A.D.: Self-Protecting Electronic Medical Records using Attribute-Based Encryption on Mobile Device. Technical report. Cryptology ePrint Archive, Report 2010/565 (2010)
27. Beimel, A.: Secure schemes for secret sharing and key distribution. PhD thesis, Israel Institute of Technology. Technion, Haifa, Israel (1996)
28. Jung, T., Li, X., Wan, Z., Wan, M.: Privacy preserving cloud data access with multi-authorities. In: Proceedings of the 32th IEEE International Conference on Computer Communications (INFOCOM'13) pp. 2625–2633 (2013)
29. Boneh, D., Boyen, X.: Efficient selective-ID secure identity based encryption without random oracles. In: Advances in Cryptology—EUROCRYPT 2004, LNCS 3027, pp. 223–238 (2004)
30. Xiao, M., Yuan, S.: Achieving fine-grained access control and integrity auditing in cloud storage. J. Comput. Inf. Syst. **9**, 5477–5484 (2013)
31. Fiore, D., Gennaro, R.: Publicly verifiable delegation of large polynomials and matrix computations, with applications. In: Proceedings of the 19th ACM Conference on Computer and Communications Security (CCS'12) pp. 501–512 (2012)
32. Zheng, Q., Xu, S., Ateniese, G.: VABKS: verifiable attribute-based keyword search over outsourced encrypted data. IACR Cryptology ePrint Archive 462 (2013)