Verifiable Attribute-based Keyword Search Encryption with Attribute Revocation for Electronic Health Record System

Zhenhua Liu[1], Yan Liu[1], Jing Xu[1], and Baocang Wang[2]
(Corresponding author: Yan Liu)

School of Mathematics and Statistics, Xidian University[1]
Xi'an 710071, P. R. China
State Key Laboratory of Integrated Services Networks, Xidian University[2]
(Email: ly10_xidian@163.com)

(Received Apr. 7, 2019; Revised and Accepted Dec. 1, 2019; First Online Jan. 29, 2020)

Abstract

Considering the security requirements of electronic health record (EHR) system, we propose a ciphertext-policy attribute-based encryption scheme, which can support data retrieval, result verification and attribute revocation. In the proposed scheme, we make use of the BLS signature technique to achieve result verification for attribute-based keyword search encryption. In addition, key encrypting key (KEK) tree and re-encryption are utilized to achieve efficient attribute revocation. By giving thorough security analysis, the proposed scheme is proven to achieve: 1) Indistinguishability against selective ciphertext-policy and chosen plaintext attack under the decisional q-parallel bilinear Diffie-Hellman exponent hardness assumption; 2) Indistinguishability against chosen-keyword attack under the bilinear Diffie-Hellman assumption in the random oracle model. Moreover, the performance analysis results demonstrate that the proposed scheme is efficient and practical in electronic health record system.

Keywords: Attribute-Based Encryption; Attribute Revocation; Electronic Health Records; Keyword Search; Verifiability

1 Introduction

Electronic health record (EHR) system can provide health record storage service that allows patients to store, manage and share their EHR data with intended clients [13]. With the development of electronic health record system, much sensitive information from patients is being uploaded into the cloud. Since the cloud server may be dishonest, it is of vital importance to protect the confidentiality of the sensitive EHR data. Furthermore, it remains to be solved that how to securely share and search EHR data without revealing the information of patients.

Traditional public key encryption can only support "one-to-one" model, which is not suitable for multi-client data sharing in EHR scenarios. Fortunately, Sahai and Waters [16] first proposed the concept of attribute-based encryption (ABE) in 2005, which can provide "one-to-many" service and be considered as one of the most appropriate encryption technologies for cloud storage. Attribute-based encryption contains two variants: Ciphertext-Policy ABE (CP-ABE), where the ciphertext is associated with access policy, and key-policy ABE (KP-ABE), where a client's secret key is associated with access policy. Furthermore, Narayan *et al.*, [12] proposed a privacy preserving EHR system using attribute-based encryption technology in 2010, which enables patients to share their data among health care providers in a flexible, dynamic and scalable manner. Li *et al.*, [9] designed a new ABE scheme for personal health records system using multi-authority ABE, which avoids the key escrow problem. Reedy *et al.*, [14] proposed a secure framework for ensuring EHR's integrity, and solved the key escrow issue by using two-authority key generation scheme. Since then, some attribute-based encryption schemes [5, 8] for EHR system have been presented.

Although attribute-based encryption can achieve fine-grained data sharing, there are many problems to be considered in practical applications. For example, when a client leaves the system or discloses the secret key, it is essential to revoke the client's attributes or secret key. In order to solve the problem, a lot of revocable ABE schemes (RABE) [3, 17, 23] have been put forward. Yu *et al.*, [25] presented a revocable CP-ABE scheme by using proxy re-encryption, which allows an untrusted server to update a ciphertext into a new ciphertext without decryption. Hur *et al.*, [7] proposed an attribute-based access control scheme with efficient revocation in data outsourcing system using key encrypting key (KEK) tree. By using Chinese remainder theorem, Zhao *et al.*, [26] introduced an efficient and revocable CP-ABE scheme in cloud computing. However, there is few RABE schemes for electronic health record system.

In addition, attribute-based encryption can protect data confidentiality, but hinder data retrieval from encrypted data in cloud storage. To address this issue, searchable encryption (SE) is proposed. SE contains two types: symmetric searchable encryption (SSE) and asymmetric searchable encryption (ASE). Song *et al.*, [18] first proposed the concept of symmetric searchable encryption. Boneh *et al.*, [1] introduced the first public-key encryption with keyword search (PEKS) scheme, and formalized a well-defined security notion of semantic security under chosen-keyword attack. After that, a lot of searchable encryption schemes [6, 15, 21] have been proposed. Furthermore, searchable encryption has widely been used in electronic health record system. For example, Xhafa *et al.*, [24] presented an efficient fuzzy keyword search scheme with multi-user over encrypted EHR data. Florence *et al.*, [4] proposed an enhanced secure sharing of personal health record system scheme with keyword search in cloud.

Searchable encryption allows a client to search over the encrypted data in cloud storage to retrieve the interested data without decryption. Nevertheless, the semi-trusted cloud server maybe performs search operation on the encrypted data and only returns a fraction of the results. In order to resist the cloud server's dishonest behavior, the verification technique [20] was introduced. Zheng *et al.*, [27] proposed a verifiable attribute-based keyword search scheme using bloom filter and digital signature techniques, which has good performance in search efficiency, but needs huge computational overhead in the verification process. Sun *et al.*, [19] introduced a verifiable attribute-based keyword search with fine-grained owner-enforced search authorization in the cloud, but the verification efficiency is low. Furthermore, Miao *et al.*, [10] proposed a verifiable multi-keyword search over the encrypted cloud data for dynamic data-owner.

Unfortunately, the above existing schemes can not achieve fine-grained access control with attribute revocation, data retrieval and result verification for EHR system, simultaneously.

1.1 Our Contributions

Based on Waters' scheme [22], we will propose a verifiable attribute-based keyword search encryption scheme with attribute revocation (VABKS-AR) for electronic health record system. Our contributions are described as follows:

1) We can achieve efficient attribute-level revocation by using a KEK tree and re-encryption. A KEK tree is utilized to distribute attribute group key and re-encryption assures that the updated ciphertext cannot be decrypted by the revoked clients.

2) Since the cloud service provider is semi-trusted, the result verification mechanism [10] is used to achieve the verifiability for attribute-based keyword search encryption, which can reduce the computational overhead of the client.

3) We provide thorough analysis of the security and performance of the proposed secure EHR sharing system. The performance analysis results show that the proposed scheme is efficient and practical for electronic health record system.

1.2 Organization

The rest of this paper is organized as follows. We describe some preliminaries and system architecture in Section 2. A formal definition and security model are given in Section 3. The proposed VABKS-AR scheme is presented in Section 4. The security proof and performance analysis are given in Section 5. Finally, we make the conclusions in Section 6.

2 Preliminaries and System Architecture

2.1 Bilinear Map

Let $\mathbb{G}$ and $\mathbb{G}_T$ be groups of prime order p, and g be a generator of $\mathbb{G}$. The map $\hat{e}$: $\mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is said to be an admissible map if it satisfies the following properties [22]:

1) **Bilinearity**: $\hat{e}(g^a, g^b) = \hat{e}(g, g)^{ab}$ for all $a, b \in \mathbb{Z}_p$.

2) **Non-degeneracy**: $\hat{e}(g, g) \neq 1$.

3) **Computability**: There is an efficient polynomial-time algorithm to compute $\hat{e}(g, g)$.

2.2 Access Structure

Let $P_1, P_2, \cdots, P_n$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{P_1, P_2, \cdots, P_n}$ is monotone for $\forall B, C$: if $B \in A$ and $B \subseteq C$, then $C \in A$. An access structure [22] (respectively, monotone access structure) is a collection (respectively, monotone collection) $\mathbb{A}$ of non-empty subsets of $P_1, P_2, \cdots, P_n$, i.e. $\mathbb{A} \subseteq 2^{P_1, P_2, \cdots, P_n} \backslash \{\emptyset\}$. The sets in $\mathbb{A}$ are called the authorized sets, and the sets not in $\mathbb{A}$ are called the unauthorized sets.

2.3 Linear Secret Sharing Scheme (LSSS)

A linear secret-sharing scheme [22] Π over a set of parties P is described as follows:

1) The shares of each party form a vector over $\mathbb{Z}_p$.

2) There exists a share-generating matrix M for Π, where M has ℓ rows and n columns. For all $i = 1, 2, \cdots, \ell$, the function ρ labels the i-th row of M as $\rho(i)$. Consider the vector $\vec{v} = (s, r_2, \cdots, r_n)$, where $s \in \mathbb{Z}_p$ is a secret to be shared, and $r_2, \cdots, r_n \in \mathbb{Z}_p$

are chosen at random. $\mu_i = M_i \cdot \vec{v}$ is one of ℓ shares of the secret s according to Π, where $M_i \in \mathbb{Z}_p^n$ is the i-th row of the matrix M. The share $M_i \cdot \vec{v}$ belongs to party $\rho(i)$.

Linear reconstruction property [22]: Suppose that Π is an LSSS for the access structure $\mathbb{A}$. Let $S \in \mathbb{A}$ be any authorized set, and $I = \{i : \rho(i) \in S\} \subseteq \{1, 2, \cdots, \ell\}$. Then, there exist constants $\{\omega_i \in \mathbb{Z}_p\}_{i \in I}$ such that, if $\{\mu_i\}$ are valid shares of any secret s according to Π, we have $\Sigma \omega_i \mu_i = s$. These constants ω_i can be found in polynomial time in the size of the share-generating matrix M.

2.4 Bilinear Diffie-Hellman (BDH) Assumption

Let $\mathbb{G}$ and $\mathbb{G}_T$ be multiplicative cyclic groups with prime order p, and g be a generator of $\mathbb{G}$. Given a tuple $\vec{y} = (g, g^a, g^b, g^c)$, where a, b, c are selected from $\mathbb{Z}_p$ randomly. The Bilinear Diffile-Hellman (BDH) problem [1] is to compute $\hat{e}(g, g)^{abc} \in \mathbb{G}_T$. An algorithm $\mathcal{B}$ has at least advantage ε in solving the Bilinear Diffile-Hellman (BDH) problem if

$$\Pr[\hat{e}(g, g)^{abc} \leftarrow \mathcal{B}(\vec{y})] \geq \varepsilon.$$

BDH Assumption: We say the BDH assumption [1] holds if no probabilistic polynomial time algorithm can solve the BDH problem with a non-negligible probability ε.

2.5 Decisional Parallel Bilinear Diffie-Hellman Exponent Assumption

Let $\mathbb{G}$ be a group with order p, and g be a generator of $\mathbb{G}$. Given

$$\begin{aligned}\vec{y} = \Big(& g, g^s, g^a, \cdots, g^{a^q}, g^{a^{q+2}}, \cdots, g^{a^{2q}}, \\ & \forall_{1 \leq j \leq q},\ g^{s \cdot b_j}, g^{a/b_j}, \cdots, g^{a^q/b_j}, g^{a^{q+2}/b_j}, \cdots, g^{a^{2q}/b_j}, \\ & \forall_{1 \leq k, j \leq q, k \neq j},\ g^{a \cdot s \cdot b_k / b_j}, \cdots, g^{a^q \cdot s \cdot b_k / b_j}\Big),\end{aligned}$$

where $a, s, b_1, \cdots, b_q \in \mathbb{Z}_p$ are chosen randomly, the decisional q-parallel bilinear Diffie-Hellman exponent (BDHE) problem [22] is to distinguish a valid tuple $\hat{e}(g, g)^{a^{q+1} \cdot s} \in \mathbb{G}_T$ from a random element $R \in \mathbb{G}_T$. An algorithm $\mathcal{B}$ has advantage ε in solving the q-parallel BDHE problem if

$$\left| \Pr[\mathcal{B}(\vec{y}, \hat{e}(g, g)^{a^{q+1} s}) = 0] - \Pr[\mathcal{B}(\vec{y}, R) = 0] \right| \geq \varepsilon.$$

Decisional q-Parallel BDHE Assumption: We say the decisional q-parallel BDHE assumption [22] holds if no probabilistic polynomial time algorithm can solve the decisional q-parallel BDHE problem with a non-negligible probability ε.

2.6 KEK Tree

Let $\mathcal{U} = \{u_1, u_2, \cdots, u_n\}$ be the universe of clients and $\mathcal{L}$ be the universe of descriptive attributes in the system. Let $G_j \subset \mathcal{U}$ be s set of clients that hold the attribute λ_j $(j = 1, 2, \cdots, q)$, which is referred to as an attribute group. G_j will be used as a client access list to λ_j. Let $\mathcal{G} = \{G_1, G_2, \cdots, G_q\}$ be the universe of attribute groups and GK_{λ_j} be the attribute group key that is shared among the non-revoked clients in $G_j \in \mathcal{G}$.

In a KEK tree [7], each node holds a KEK_j. A set of KEKs on the path node from leaf to root are called the path keys. A KEK tree is constructed by the data service manager as follows:

1) Each client u_{id} $(id = 1, 2, \cdots, n)$ in the universe $\mathcal{U}$ is assigned to a leaf node of the tree. Random keys are generated and assigned to all leaf nodes and internal nodes.

2) Each client $u_{id} \in \mathcal{U}$ obtains the path keys PAK_{id} from its leaf node to the root node of tree, securely. For example, the client u_4 has the path keys $PAK_4 = \{KEK_{11}, KEK_5, KEK_2, KEK_1\}$ in Figure 1.

3) The minimum cover sets [11] $node(G_j)$ is a minimum set of nodes in the tree, which can cover all of the leaf nodes associated with clients in G_j. $KEK(G_j)$ is a set of KEK values owned by $node(G_j)$. To consider the intersection of PAK_{id} and $KEK(G_j)$, we have $KEK = KEK(G_j) \cap PAK_{id}$.

Let us give an example to illustrate the attribute groups G_j. Suppose $\{u_1, u_2, u_3\}$ are associated with $\{\lambda_1, \lambda_2\}, \{\lambda_1, \lambda_2, \lambda_3\}, \{\lambda_2, \lambda_3\}$, respectively. We have the attribute group $G_1 = \{u_1, u_2\}, G_2 = \{u_1, u_2, u_3\}, G_3 = \{u_2, u_3\}$.

Consider the example in Figure 1. If the attribute group for attribute λ_j is $G_j = \{u_2, u_3, u_5, u_6, u_7, u_8\}$ and u_6 is associated with leaf node v_{13}, we compute the minimum cover sets $node(G_j) = \{v_9, v_{10}, v_3\}$ and get $KEK(G_j) = \{KEK_9, KEK_{10}, KEK_3\}$, which will be used to encrypt the attribute group key GK_{λ_j} in the data re-encryption phase. Since u_6 stores path keys $PAK_6 = \{KEK_{13}, KEK_6, KEK_3, KEK_1\}$, we have $KEK = KEK(G_j) \cap PAK_6 = \{KEK_3\}$, then u_6 can decrypt the header message to get the attribute group key GK_{λ_j} using KEK_3.

2.7 System Architecture

As shown in Figure 2, a verifiable attribute-based keyword search encryption scheme with attribute revocation (VABKS-AR) system consists of five entities: Trusted Authority (TA), Cloud Service Provider (CSP), Data Owner/Patient, Client/Doctor, and Third Party Audit (TPA).

- **Trusted Authority (TA)**: TA generates the public parameter, the master secret key and the clients'

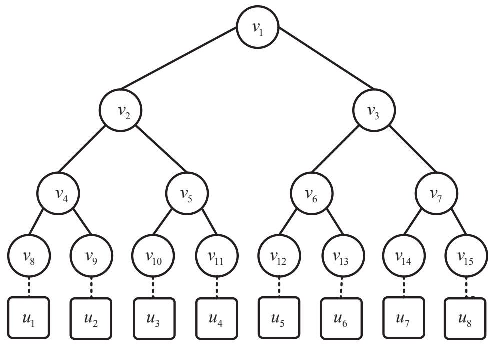

Figure 1: KEK tree

secret key according to their attributes. TA is fully trusted in the system.

- **Cloud Service Provider (CSP)**: CSP consists of a data server and a data service manager. The data server has huge storage space and a strong computational power. The data service manager is in charge of managing the attribute group keys of each attribute group and providing the corresponding services. We assume the data service manager is honest-but-curious. i.e., it will honestly performs the operation but try to acquire much more information about the sensitive data.

- **Data Owner/Patient**: A patient is viewed as the data owner, which encrypts the sensitive data (i.e. electronic health record system data) and the keyword, and then uploads them to CSP in the form of ciphertext. Meanwhile, the data owner enforces the access policy for encrypted data, where the ciphertext will be shared with the client whose attributes satisfy the access structure embedded in ciphertext.

- **Client/Doctor**: A doctor is viewed as a client, which submits a search query to retrieve the encrypted EHR stored on the cloud server. Upon receiving the query, the cloud server searches the intended ciphertext by the use of trapdoor. If a client is not revoked and her or his attribute set satisfies the access policy, the client can decrypt the ciphertext.

- **Third Party Audit (TPA)**: TPA can provide the verification of search result and response a challenge to CSP. Upon receiving the challenge, CSP returns a proof to TPA. Finally, TPA calculates a value to verify the integrity of returned search results.

3 Formal Definition and Security Model

3.1 Formal Definition

In this section, the formal definition of a verifiable attribute-based keyword search encryption with attribute

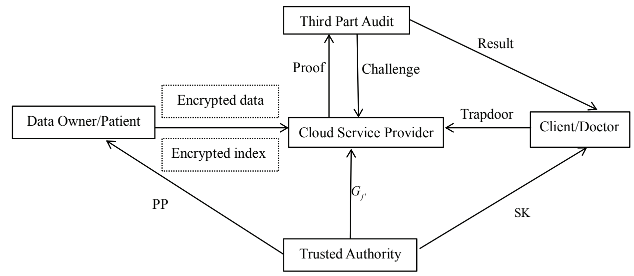

Figure 2: System architecture of VABKS-AR

revocation is described as the following nine algorithms:

- **Setup**$(1^\lambda) \rightarrow (PP, MSK)$: TA takes a security parameter λ as input, and outputs the public parameter PP and a master secret key MSK.

- **KeyGen**$(MSK, id, S) \rightarrow SK$: TA runs this algorithm, which takes the master secret key MSK, the identifier id of a legal client u_{id} and attribute set $S \subseteq \mathcal{L}$ as input. This algorithm outputs a secret key SK to the client u_{id}.

- **Encrypt**$(PP, (M, \rho), \mathcal{K}, W) \rightarrow (CT, I_W)$: The data owner runs this algorithm, which takes the public parameter PP, an access policy (M, ρ), the symmetric key $\mathcal{K}$, and a set of keyword W as input. Using key encapsulation technology, this algorithm outputs a ciphertext CT and an encrypted index set I_W.

- **Re-encrypt**$(CT, G, RL) \rightarrow (CT', \hat{C})$: This algorithm is performed by the data service manager. Taking the ciphertext CT, attributes group $G \subseteq \mathcal{G}$ and a revocation list RL as input. This algorithm outputs a re-encrypted ciphertext CT' and a header message $\hat{C}$.

- **Trapdoor**$(SK, w) \rightarrow tk$: The client runs this algorithm, which takes a secret key SK and a keyword w as input. This algorithm outputs tk to CSP.

- **Search**$(PP, tk, I_W) \rightarrow (C', ID')$: CSP runs this algorithm, which takes the public parameter PP, a search token tk and an encrypted index set I_W as input. This algorithm outputs intended encrypted file set C' and corresponding identifier ID' to TPA if the search token tk matches with the index set I_W; otherwise, outputs $\perp$.

- **Verify**$(PK, C', ID') \rightarrow (0, 1)$: TPA runs this algorithm, which takes the data owner's PK, the returned encrypted file set C' and corresponding identifier set ID' as input. This algorithm outputs 1 if passes the result verification; otherwise, outputs 0.

- **Decrypt**$(CT', SK) \rightarrow \mathcal{K}$: The client runs this algorithm, which takes the ciphertext CT' and a secret key SK as input. This algorithm outputs the symmetric key $\mathcal{K}$.

- **CTUpdate**$(CT', RL') \to (CT'', \hat{C}')$: The data service manager runs this algorithm, which takes the re-encrypted ciphertext CT' and a new revocation list RL' as input. This algorithm outputs an updated ciphertext CT'' and a new header message $\hat{C}'$.

3.2 Security Model

In this section, we will give two security models: indistinguishability against selective ciphertext-policy and chosen plaintext attack (IND-sCP-CPA) game and indistinguishability against chosen keyword attack (IND-CKA) game. The security of our scheme is based on the following two games:

Firstly, according to Waters' scheme [22], we describe the **IND-sCP-CPA game** as follows:

- **Init.** The adversary $\mathcal{A}$ gives the challenge access policy (M^*, ρ^*) and a revocation list RL^*, where M^* has $n^* \le q$ columns.
- **Setup.** The challenger $\mathcal{B}$ runs the *Setup* algorithm, sends the public parameter PP to $\mathcal{A}$, and then keeps the master secret key MSK for himself.
- **Phase 1.** The adversary $\mathcal{A}$ issues polynomial time secret key queries for (id, S). The challenger $\mathcal{B}$ sends SK to the adversary A, but with the restriction that:
  1) if $u_{id} \notin RL^*$, $S' = S$, and the set of attribute S' does not satisfy the challenge access policy (M^*, ρ^*).
  2) if $u_{id} \in RL^*$, then $S' = S \setminus \{\lambda_{j^*}\}$, and the set of attribute S' does not satisfy the challenge access policy (M^*, ρ^*).
- **Challenge.** The adversary $\mathcal{A}$ selects two equal length message k_0 and k_1 to the challenger $\mathcal{B}$. Then $\mathcal{B}$ randomly selects one bit $b \in \{0,1\}$ and encrypts k_b under (M^*, ρ^*) and the revocation list RL^*. Finally, $\mathcal{B}$ sends the challenge ciphertext CT^* to $\mathcal{A}$.
- **Phase 2.** Same as Phase 1.
- **Guess.** The adversary $\mathcal{A}$ outputs its guess b' of b and wins the game if $b' = b$.

The advantage of the adversary $\mathcal{A}$ is defined as follows:

$$Adv_{\mathcal{A}}^{IND-sCP-CPA} = \left| \Pr[b' - b] - \frac{1}{2} \right|.$$

Definition 1. *A verifiable attribute-based keyword search encryption scheme with attribute revocation is IND-sCP-CPA secure if all polynomial time adversaries have at most a negligible advantage in the above game.*

Secondly, according to Boneh's scheme [1], we define the **IND-CKA game** as follows:

- **Setup.** The challenger $\mathcal{B}$ runs the Setup algorithm, sends the public parameter PP to $\mathcal{A}$, and then keeps the master secret key MSK for himself.
- **Phase 1.** The adversary $\mathcal{A}$ can adaptively query the challenger $\mathcal{B}$ for the trapdoor T_w of any keyword $w \in \{0,1\}^*$ in polynomial time.
- **Challenge.** The adversary $\mathcal{A}$ sends two equal length keywords w_0 and w_1 to the challenger $\mathcal{B}$. The only restriction is that w_0 and w_1 have not been queried for the trapdoor. The challenger $\mathcal{B}$ randomly selects one bit $b \in \{0,1\}$, generates index I_{w_b} for keyword w_b, and submits the challenge index I_{w_b} to the adversary $\mathcal{A}$.
- **Phase 2.** The adversary $\mathcal{A}$ can issue more trapdoor queries for keyword w with the restriction $w \neq w_0, w_1$.
- **Guess.** The adversary $\mathcal{A}$ outputs its guess b' of b and wins the game if $b' = b$.

The advantage of the adversary $\mathcal{A}$ is defined as follows:

$$Adv_{\mathcal{A}}^{IND-CKA} = \left| \Pr[b' - b] - \frac{1}{2} \right|.$$

Definition 2. *A verifiable attribute-based keyword search encryption scheme with attribute revocation is IND-CKA secure if all polynomial time adversaries have at most a negligible advantage in the above game.*

4 Concrete Construction

The concrete construction is described as follows:

- **Setup**(1^λ): This algorithm selects a bilinear map $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, such that $\mathbb{G}$ and $\mathbb{G}_T$ are cyclic groups of order p, an λ-bit prime, and $E(\cdot)$ be a probabilistic symmetric encryption algorithm. We define three hash functions $H : \mathbb{Z}_p \to \mathbb{G}, H_1 : \{0,1\}^* \to \mathbb{G}, H_2 : \mathbb{G}_T \to \{0,1\}^{\log_2 p}$. Let CL be a client list and RL be a revocation list, where CL and RL are initially empty. TA runs this algorithm as follows:
  1) Pick random $a, \alpha \in \mathbb{Z}_p$ and compute
  $$h = g^a, Y = \hat{e}(g,g)^\alpha.$$
  2) Publish the public parameter
  $$PP = \big(\hat{e}, h, Y, H, H_1, H_2, E(\cdot), CL, RL\big)$$
  and keep the master secret key $MSK = \alpha$ himself.
- **KeyGen**(MSK, id, S): A client sends its identifier id and a set of attributes $S \subseteq \mathcal{L}$ to TA. TA runs this algorithm as follows:
  1) Select $t \in \mathbb{Z}_p$ randomly and compute a secret key
  $$SK = \big(K = g^\alpha g^{at}, L = g^t, \{K_j = H(\lambda_j)^t\}_{\lambda_j \in S}\big)$$
  for the client $u_{id} \in \mathcal{U}$.

2) Add (id, g^{at}) to the client list CL and send SK to the client.

- **Encrypt**$(PP, (M, \rho), \mathcal{K}, \mathcal{F}, W)$: The data owner inputs the public parameter PP, an access policy (M, ρ), a symmetric key $\mathcal{K}$, a set of data file $\mathcal{F}$ and a set of keyword W. This algorithm is run by the data owner as follows:

  1) Encrypt the data file $\mathcal{F} = (f_1, f_2, \cdots f_d)$ as $c_k = E_{\mathcal{K}}(f_k)$ $(1 \leq k \leq d)$ with the symmetric key $\mathcal{K}$.
  2) Select random $s, y_2, \cdots, y_n \in \mathbb{Z}_p$ and set a column vector $\vec{v} = (s, y_2, \cdots, y_n)$. For $1 \leq i \leq \ell$, compute $\mu_i = M_i \cdot \vec{v}$, where M_i is the i-th row of M. Choose random numbers $r_1, \cdots, r_\ell \in \mathbb{Z}_p$ and calculate
  $$CT = \{C_{0,0} = \mathcal{K} \cdot \hat{e}(g,g)^{\alpha s}, C_{0,1} = g^s, C_{0,2} = h^s, \forall i = 1, \cdots, \ell : C_i = g^{a\mu_i} H(\rho(i))^{-r_i}, D_i = g^{r_i}\}.$$
  3) Extract a set of keywords
  $$W = (w_1, w_2, \cdots, w_m)$$
  from the data files $\mathcal{F}$. For each keyword w_δ $(1 \leq \delta \leq m)$, compute
  $$\varphi_\delta = \hat{e}(g,g)^{\alpha s} \cdot \hat{e}(g, H_1(w_\delta))^s$$
  and
  $$I_W = \{I_{w_\delta} = H_2(\varphi_\delta)\}_{\delta=1}^m,$$
  where I_W is the encrypted index set for the keyword set W.
  4) Select a random $x \in \mathbb{Z}_p$, and compute $PK = g^x$ as its public key. For each encrypted data file c_k with identifier k, calculate a signature $\sigma_k = (H_1(k) g^{c_k})^x$ with the data owner's secret key x.

  After the construction of CT, the data owner sends $(CT, I_W, \{c_k, \sigma_k\}_{k=1}^d)$ to CSP.

- **Re-encrypt**(CT, G, RL): This algorithm inputs a ciphertext CT, a set of attribute group $G \subseteq \mathcal{G}$ and a revocation list RL. The data service manager runs this algorithm as follows:

  1) For $\forall\, G_j \in G$, choose a random attribute group key $GK_{\lambda_j} \in \mathbb{Z}_p^*$, and re-encrypt CT as:
  $$\begin{aligned} CT' = \{ & C'_{0,0} = C_{0,0}, C'_{0,1} = C_{0,1}, C'_{0,2} = C_{0,2}, \\ & \forall\, i = 1, \cdots, \ell : C'_i = C_i, \\ & RL = \emptyset : D'_i = D_i, \\ & RL \neq \emptyset : D'_i = {D_i}^{GK_{\lambda_j}}\}. \end{aligned}$$
  2) Compute the minimum cover sets $node(G_j)$ of G_j in the KEK tree, get the corresponding $KEK(G_j)$, and generate a ciphertext $\hat{C} = \{E_\kappa(GK_{\lambda_j})\}_{\kappa \in KEK(G_j)}$, which called the header message.

- **Trapdoor**(SK, w): A client with identifier id and attribute set S inputs a secret key SK and a keyword w, The algorithm runs as follows:

  1) The client selects $u \in \mathbb{Z}_p$ randomly, computes $q_u = g^{1/u}$, and sends (id, q_u) to TA. Then, TA retrieves g^{at} according to id in the client list CL, computes $q_{id} = g^{at} q_u^\alpha$, and sends q_{id} to the client.
  2) The client calculates a search token $tk = \big(T_w = H_1(w) q_{id}^u, L' = L^u, \{K'_j = K_j^u\}_{\lambda_j \in S}\big)$ and sends tk to the data service manager.

- **Search**(PP, tk, I_W): This algorithm inputs the public parameter PP, a search token tk, and encrypted index set I_W. CSP runs this algorithm as follows:

  1) Compute
  $$l_w = \frac{\hat{e}(C'_{0,1}, T_w)}{\hat{e}(L', C'_{0,2})} = \hat{e}(g,g)^{\alpha s} \cdot \hat{e}\big(g, H_1(w)\big)^s.$$
  2) If there exists some encrypted index I_{w_δ} such that $H_2(l_w) = I_{w_\delta}$, send $(CT', \hat{C})$, the relevant encrypted file set $C' = \{c_1, c_2, \cdots, c_\tau\}$ and the corresponding identifier set $ID' = \{1, 2 \cdots, \tau\}$ to TPA, where τ is the number of returned files; otherwise, return $\perp$.

- **Verify**(PK, C', ID'): This algorithm inputs the data owner's PK, the returned encrypted file set C' and corresponding identifier set ID'. TPA runs this algorithm as the following steps:

  1) TPA randomly selects $v_r \in \mathbb{Z}_p$, and generates a $chal = < r, v_r >$ $(r \in [1, \tau])$ to CSP.
  2) Upon receiving the $chal$ of TPA, CSP computes $\zeta = \Sigma_{r \in [1,\tau]} v_r c_r$ and $\sigma = \Pi_{r \in [1,\tau]} {\sigma_r}^{v_r}$, where $\sigma_r = (H_1(r) g^{c_r})^x$. Then CSP sends (ζ, σ) to TPA.
  3) TPA verifies whether the following equation holds or not. If hold, return 1 and send $(CT', \hat{C}, C', ID')$ to the client; otherwise, return 0.
  $$\hat{e}(\sigma, g) = \hat{e}\big(g^\zeta \cdot \Pi_{r \in [1,\tau]} H_1(r)^{v_r}, PK\big).$$

- **Decrypt**(CT', SK): This algorithm inputs the ciphertext CT', a secret key SK, and runs as follows:

  1) If a client has a valid attribute λ_j, i.e. $u_{id} \in G_j$, he can use a $KEK \in \big(KEK(G_j) \cap PAK_{id}\big)$ to get the attribute group key GK_{λ_j}. And then u_{id} updates its secret key with the attribute group keys as follows:
  $$\begin{aligned} SK = \big(& K = g^\alpha g^{at}, L = g^t, \\ & \{K_j = H(\lambda_j)^{t/GK_{\lambda_j}}\}_{\lambda_j \in S}\big). \end{aligned}$$
  2) Output $(0, \perp)$ if S does not satisfy (M, ρ).

3) Otherwise, let $I \subset \{1, 2, \cdots, \ell\}$ be defined as $I = \{i : \rho(i) \in S\}$ and $\{\omega_i \in \mathbb{Z}_p | i \in I\}$ be a set of constants such that if μ_i are valid shares of any secret s according to M, then $\Sigma\omega_i\mu_i = s$. We have

$$Q_{CT} = \prod_{i\in I} \Big(\hat{e}(C'_i, L) \cdot \hat{e}(D'_i, K'_j)\Big)^{\omega_i} = \hat{e}(g, g)^{ast}.$$

4) Decrypt the ciphertext and obtain the symmetric key: $\mathcal{K} = C'_{0,0} \cdot \frac{Q_{CT}}{\hat{e}(C'_{0,1}, K)}$.

5) Decrypt the encrypted data files C' using $\mathcal{K}$.

- **CTUpdate**(CT', RL'): This algorithm inputs CT' and a new revocation list RL'. If an attribute $\lambda_{j'}$ of the client is revoked, TA sends the updated membership list $G_{j'}$ to CSP. The data service manager runs this algorithm as follows:

1) Select random $s', y'_2, \cdots, y'_n \in \mathbb{Z}_p^n$, a new attribute group key $GK'_{\lambda_{j'}}$, and set a column vector $\vec{v'} = (s', y'_2, \cdots, y'_n) \in \mathbb{Z}_p^n$. For $1 \le i \le \ell$, compute $\mu'_i = M_i \cdot \vec{v'}$, where M_i is the i-th row of M.

2) Choose random numbers $r'_1, \cdots, r'_\ell \in \mathbb{Z}_p$ and update the ciphertext CT' as:

$$\begin{aligned} CT'' = \big(&C''_{0,0} = C'_{0,0} \cdot \hat{e}(g, g)^{\alpha s'}, \\ &C''_{0,1} = C'_{0,1} \cdot g^{s'}, C''_{0,2} = C'_{0,2} \cdot h^{s'}, \\ &\forall\, i = 1, \cdots, \ell : C''_i = C'_i \cdot g^{a\mu'_i} H(\rho(i))^{-r'_i}, \\ &\rho(i) \in RL' : D''_i = D'_i \cdot (g^{r'_i})^{GK'_{\lambda_{j'}}}, \\ &\rho(i) \notin RL' : D''_i = D'_i \cdot (g^{r'_i})^{GK_{\lambda_j}}\big). \end{aligned}$$

3) Compute a new minimum cover set and generate a new header message with updated $KEK(G_{j'})$ as follows:

$$\begin{aligned} \hat{C}' =& (\{E_\kappa(GK'_{\lambda_{j'}})\}_{\kappa\in KEK(G_{j'})}, \\ & \forall_{\lambda_j \in S\backslash\{\lambda_{j'}\}} : \{E_\kappa(GK_{\lambda_j})\}_{\kappa\in KEK(G_j)}). \end{aligned}$$

Correctness. The proposed scheme is correct as the following equations hold:

$$\begin{aligned} l_w &= \frac{\hat{e}\big(C'_{0,1}, T_w\big)}{\hat{e}(L', C'_{0,2})} = \frac{\hat{e}\big(g^s, H_1(w) g^{atu} g^{\alpha}\big)}{\hat{e}\big((g^t)^u, g^{as}\big)} \\ &= \frac{\hat{e}(g,g)^{\alpha s} \cdot \hat{e}\big(g, H_1(w)\big)^s \cdot \hat{e}(g,g)^{astu}}{\hat{e}(g,g)^{astu}} \\ &= \hat{e}(g,g)^{\alpha s} \cdot \hat{e}\big(g, H_1(w)\big)^s \end{aligned}$$

$$\begin{aligned} Q_{CT} &= \prod_{i\in I}\Big(\hat{e}(C'_i, L) \cdot \hat{e}(D'_i, K_j)\Big)^{\omega_i} \\ &= \prod_{i\in I}\Big(\hat{e}\big(g^{a\mu_i} H\big(\rho(i)\big)^{-r_i}, g^t\big) \\ &\quad \cdot \hat{e}\big((g^{r_i})^{GK_{\lambda_j}}, H\big(\rho(i)\big)^{t/GK_{\lambda_j}}\big)\Big)^{\omega_i} \\ &= \hat{e}(g,g)^{\Sigma a\mu_i\omega_i t} = \hat{e}(g,g)^{ast} \\ \mathcal{K} &= C'_{0,0} \cdot \frac{Q_{CT}}{\hat{e}(C'_{0,1}, K)} \\ &= \mathcal{K} \cdot \hat{e}(g,g)^{\alpha s} \frac{\hat{e}(g,g)^{ast}}{\hat{e}(g^s, g^{\alpha} g^{at})} \\ &= \mathcal{K} \cdot \hat{e}(g,g)^{\alpha s} \frac{\hat{e}(g,g)^{ast}}{\hat{e}(g^s, g^{\alpha}) \cdot \hat{e}(g,g)^{ast}} \\ &= \mathcal{K} \cdot \hat{e}(g,g)^{\alpha s} \frac{1}{\hat{e}(g,g)^{\alpha s}} = \mathcal{K} \\ \hat{e}(\sigma, g) &= \hat{e}\big(\Pi_{r\in[1,\tau]}\sigma_r^{v_r}, g\big) \\ &= \hat{e}\Big(\Pi_{r\in[1,\tau]}\big(H_1(r) g^{c_r}\big)^{xv_r}, g\Big) \\ &= \hat{e}\Big(\Pi_{r\in[1,\tau]}\big(H_1(r)^{v_r} \cdot g^{\Sigma v_r c_r}\big)^x, g\Big) \\ &= \hat{e}\Big(\Pi_{r\in[1,\tau]} H_1(r)^{v_r} \cdot g^{\zeta}, g^x\Big) \\ &= \hat{e}\Big(g^{\zeta} \cdot \Pi_{r\in[1,\tau]} H_1(r)^{v_r}, PK\Big) \end{aligned}$$

5 Security and Performance

5.1 Security Analysis

Theorem 1. *If a probabilistic polynomial-time adversary $\mathcal{A}$ wins the IND-sCP-CPA game with non-negligible advantage ε, then we can construct a simulator $\mathcal{B}$ to solve the q-parallel BDHE problem with non-negiligible advantage $\varepsilon' = \varepsilon/2$.*

Proof. Suppose $\mathcal{A}$ is an adversary that has advantage ε in breaking the IND-sCP-CPA game. We construct a simulator $\mathcal{B}$ that can solve the q-parallel BDHE problem with probability at least ε'.

- **Init.** The simulator $\mathcal{B}$ is given a decisional q-parallel challenge vector $\vec{y}$ and a random number T. The adversary $\mathcal{A}$ selects the challenge access policy (M^*, ρ^*) and the revocation list RL^*, where M^* has $n^* \le q$ columns.

- **Setup.** The simulator $\mathcal{B}$ randomly selects $\alpha' \in \mathbb{Z}_p$, computes $\hat{e}(g,g)^{\alpha} = \hat{e}(g^a, g^{a^q}) \cdot \hat{e}(g,g)^{\alpha'}$, which implies that $\alpha = \alpha' + a^{q+1}$. We use a list called H-list to run the random oracle H for $\mathcal{B}$. The simulator $\mathcal{B}$ responds as follows:

1) If $H(j)$ has already appeared on the H-list, then $\mathcal{B}$ returns the value that was predefined before.

2) Otherwise, let X be the set of indices i that makes $\rho^*(i) = \lambda_{j^*}$ true. $\mathcal{B}$ randomly selects a number $z_{j^*} \in \mathbb{Z}_p$, and executes the random oracle:

– If $X = \emptyset$, $H(j^*) = g^{z_{j^*}}$;
– If $X \neq \emptyset$, we have

$$H(j^*) = g^{z_{j^*}} \prod_{i \in X} g^{aM^*_{i,1}/b_i} \cdot g^{a^2 M^*_{i,2}/b_i} \cdots g^{a^{n^*} M^*_{i,n^*}/b_i} \ (n^* \leq q).$$

- **Phase 1.** The adversary $\mathcal{A}$ issues polynomial time secret key queries for (id, S). Suppose the adversary sends the identifier and the corresponding set of attributes (id, S) to the simulator $\mathcal{B}$, but the following restrictions must be satisfied:

  1) If $u_{id} \notin RL^*$, $S' = S$, and the attributes set S' does not satisfy (M^*, ρ^*).
  2) If $u_{id} \in RL^*$, then $S' = S \setminus \{\lambda_{j^*}\}$, and the attributes set S' does not satisfy (M^*, ρ^*).

  The simulator $\mathcal{B}$ chooses a vector $\overrightarrow{\omega} = (\omega_1, \omega_2, \cdots, \omega_{n^*}) \in \mathbb{Z}_p^{n^*}$. For any i, such that $\rho^*(i) \in S'$ and $\omega_1 = -1$, we have $M_i^* \cdot \overrightarrow{\omega} = 0$. $\mathcal{B}$ randomly selects a number $r \in \mathbb{Z}_p$, and computes a secret key as follows:

$$K = g^{\alpha} g^{at} = g^{\alpha'} g^{ar} \prod_{i=2,\cdots,n^*} (g^{a^{q+2-i}})^{\omega_i},$$
$$L = g^t = g^r \cdot \prod_{1,\cdots,n^*} (g^{a^{q+1-i}})^{\omega_i},$$

  which implies

$$t = r + \omega_1 a^q + \omega_2 a^{q-1} + \cdots + \omega_{n^*} a^{q-n^*+1}.$$

  For $\forall\ \lambda_{j^*} \in S'$, when there is no i such that $\rho^*(i) = \lambda_{j^*}$, we let $K_j = L^{z_{j^*}}$. While for those attributes $\lambda_{j^*} \in S'$ that satisfy the access structure, $\mathcal{B}$ can not simulate the items $g^{a^{(q+1)/b_i}}$. However, we have $M_i^* \cdot \overrightarrow{\omega} = 0$. Therefore, all of these terms of $g^{a^{(q+1)/b_i}}$ can be canceled. Let X be the set of indices i such that $\rho^*(i) = \lambda_{j^*}$. The simulator $\mathcal{B}$ computes K_{j^*} as follows:

$$K_{j^*} = L^{z_{j^*}} \prod_{i \in X} \prod_{j=1,\cdots,n^*} \Big((g^{(a^j/b_i)r}) \cdot \prod_{k=1,\cdots,n^*, k \neq j} (g^{a^{q+1+j-k/b_i}})^{\omega_k} \Big)^{M^*_{i,j}}.$$

- **Challenge.** The adversary $\mathcal{A}$ chooses two equal length challenge message k_0 and k_1 to the simulator $\mathcal{B}$. $\mathcal{B}$ randomly selects a number $s \in \mathbb{Z}_p$ and a random bit $\gamma \in \{0, 1\}$. Then it computes as follows:

$$C^*_{0,0} = k_\gamma \cdot T \cdot e(g^s, g^{\alpha'}), C^*_{0,1} = g^s, C^*_{0,2} = h^s.$$

  It is difficult for $\mathcal{B}$ to simulate C_i^* since it contains $g^{a^j s}$ that $\mathcal{B}$ can not simulate. However, $\mathcal{B}$ randomly selects $y'_2, \cdots, y'_{n^*} \in \mathbb{Z}_p$ and $r'_1, \cdots, r'_\ell \in \mathbb{Z}_p$. Then simulator $\mathcal{B}$ shares the secret s utilizing the vector $\vec{v} = (s, sa + y'_2, sa^2 + y'_3, \cdots, sa^{n^*-1} + y'^*_n) \in \mathbb{Z}_p^{n^*}$. For $i = 1, \cdots, \ell$, we define R_i as the set of all $k \neq i$ such that $\rho^*(i) = \rho^*(k)$. The challenge ciphertext C_i^* is set as:

$$C_i^* = H(\rho^*(i))^{r'_i} \Big(\prod_{j=2,\cdots,n^*} (g^a)^{-M^*_{i,j} y'_j} \Big) (g^{s \cdot b_i})^{-z_{\rho^*(i)}} \cdot \Big(\prod_{k \in R_i} \prod_{j=1,\cdots,n^*} (g^{a^j \cdot s \cdot b_i / b_k}) \Big)^{-M^*_{k,j}}.$$

  1) For the non-revoked attribute $\rho^*(i)$, a challenge ciphertext is set as $D_i^* = g^{-r'_i} g^{-sb_i}$.
  2) For the revoked attribute $\rho^*(i) = \lambda_{j^*}$ and $j^* \neq i$, by selecting a random number $GK'_{\lambda_{j^*}}$, $\mathcal{B}$ computes the challenge ciphertext $D_i^* = (g^{-r'_i} g^{-sb_i})^{GK'_{\lambda_{j^*}}}$.

  $\mathcal{B}$ gives the challenge ciphertext

$$CT^* = (C^*_{0,0}, C^*_{0,1}, C^*_{0,2}, \{C_i^*, D_i^*\}_{i=1,\cdots,\ell})$$

  to $\mathcal{A}$.

- **Phase 2.** Same as Phase 1.
- **Guess.** The adversary $\mathcal{A}$ outputs γ' of γ. $\mathcal{B}$ returns $\mu = 0$ and responds $T = e(g, g)^{a^{q+1} \cdot s}$ if $\gamma' = \gamma$; otherwise, $\mathcal{B}$ returns $\mu = 1$ and responds $T \in \mathbb{G}_T$ as a random element.

If $\mu = 0, \mathcal{A}$ obtains a valid ciphertext of k_γ. The advantage of $\mathcal{A}$ in this situation is ε, therefore $\Pr[\gamma' = \gamma | \mu = 0] = 1/2 + \varepsilon$. Since $\mathcal{B}$ guesses $\mu' = 0$ when $\gamma' = \gamma$, we have $\Pr[\mu' = \mu | \mu = 0] = 1/2 + \varepsilon$.

If $\mu = 1$, we have $\Pr[\gamma' \neq \gamma | \mu = 1] = 1/2$. As $\mathcal{B}$ guesses $\mu' = 1$ when $\gamma' \neq \gamma$, we have $\Pr[\mu' = \mu | \mu = 1] = 1/2$.

The advantage of $\mathcal{B}$ to solve the decisional q-parallel BDHE problem is $\varepsilon' = \varepsilon/2$ as follows.

$$\begin{aligned} &\Pr[\mu' = \mu] - \frac{1}{2} \\ =&\frac{1}{2} \Pr[\mu' = \mu | \mu = 0] + \frac{1}{2} \Pr[\mu' = \mu | \mu = 1] - \frac{1}{2} \\ =&\frac{1}{2}(\frac{1}{2} + \varepsilon) + \frac{1}{2} \cdot \frac{1}{2} - \frac{1}{2} \\ =&\frac{\varepsilon}{2}. \end{aligned}$$

□

Theorem 2. *If a probabilistic polynomial-time adversary $\mathcal{A}$ wins the IND–CKA game with non-negligible advantage ϵ, then we can construct a simulator $\mathcal{B}$ to solve the BDH problem with non-negiligible advantage $\epsilon' = \epsilon/(e \cdot q_T \cdot q_{H_2})$ where e is the base of the nature logarithm.*

Proof. Suppose $\mathcal{A}$ is an attack algorithm that has advantage ϵ in breaking the IND–CKA game. We construct an algorithm $\mathcal{B}$ that solve the BDH problem with probability at least ϵ'.

Suppose that $\mathcal{A}$ makes at most q_{H_2} hash function queries to H_2 and at most q_T trapdoor queries. The algorithm $\mathcal{B}$ is given $g, u_1 = g^\alpha, u_2 = g^\beta, u_3 = g^\gamma \in \mathbb{G}$. It aims at outputing $\hat{e}(g,g)^{\alpha\beta\gamma} \in \mathbb{G}_T$.

- **Setup.** The algorithm $\mathcal{B}$ starts by giving $\mathcal{A}$ the public parameters PP. $\mathcal{B}$ simulates the challenger and interacts with $\mathcal{A}$ as follows:
- **Phase 1.** The adversary $\mathcal{A}$ can query the following random oracle at any time.

  $\mathcal{O}_{H_1}(w_\eta)$: The algorithm $\mathcal{B}$ creates a list of tuple $< w_\eta, h_\eta, a_\eta, c_\eta >$ called the H_1-list. The list is initially empty. When $\mathcal{A}$ asks the random oracle H_1 at the point of $w_\eta \in \{0,1\}^*$, $\mathcal{B}$ responds as follows:

  1) If the query w_η appears on the H_1-list in a tuple $< w_\eta, h_\eta, a_\eta, c_\eta >$, then $\mathcal{B}$ responds with $H_1(w_\eta) = h_\eta$.
  2) Otherwise, $\mathcal{B}$ selects a random $c_\eta \in \{0,1\}^*$ so that $\Pr[c_\eta = 0] = 1/(q_T + 1)$.
  3) $\mathcal{B}$ picks a random $a_\eta \in \mathbb{Z}_p$. If $c_\eta = 0$, $\mathcal{B}$ computes $h_\eta = u_2 \cdot g^{a_\eta}$; otherwise, $\mathcal{B}$ computes $h_\eta = g^{a_\eta}$. The algorithm $\mathcal{B}$ adds the tuple $< w_\eta, h_\eta, a_\eta, c_\eta >$ to the H_1-list and responds to $\mathcal{A}$ with $H_1(w_\eta) = h_\eta$.

  $\mathcal{O}_{H_2}(\varphi_\eta)$: The H_2-list is initially empty. At any time the adversary $\mathcal{A}$ can issue a query to H_2. The algorithm $\mathcal{B}$ responds as follows:

  1) If the query on φ_η exists in the H_2-list, $\mathcal{B}$ responds I_{w_η} to $\mathcal{A}$.
  2) Otherwise, $\mathcal{B}$ picks a new random value $I_{w_\eta} \in \{0,1\}^{\log p}$ for each new φ_η and sets $H_2(\varphi_\eta) = I_{w_\eta}$. The algorithm $\mathcal{B}$ adds the pair $(\varphi_\eta, I_{w_\eta})$ to the H_2-list and sends I_{w_η} to $\mathcal{A}$.

  $\mathcal{O}_{q_{id}}(id)$: The algorithm B creates a list of tuple $< SK, q_{id}, C >$ called the table T. Upon receiving a query of secret key on $\mathcal{A}$ and a commitment value C. $\mathcal{B}$ checks whether the tuple appears on T.

  1) If so, $\mathcal{B}$ returns q_{id} to $\mathcal{A}$.
  2) Otherwise, $\mathcal{B}$ sets $q_{id} = g^{\alpha/u} \cdot g^{at}$ and sends it to $\mathcal{A}$.

  $\mathcal{O}_{tk}(id, w_\eta)$: The adversary $\mathcal{A}$ issues a query for the trapdoor corresponding to the keyword w_η and the client identifier id, and then $\mathcal{B}$ responds as follows:

  1) $\mathcal{B}$ runs the above H_1-queries to obtain $h_\eta \in \mathbb{G}$ such that $H_1(w_\eta) = h_\eta$. Let $< w_\eta, h_\eta, a_\eta, c_\eta >$ be the corresponding tuple on the H_1-list. If $c_\eta = 0$, then $\mathcal{B}$ responds failure and aborts the game;
  2) Otherwise, we know $c_\eta = 1$ and $h_\eta = g^{a_\eta}$. $\mathcal{B}$ selects u from $\mathbb{Z}_p$ randomly, searches the table T for SK, and sets $tk = (g^{a_\eta} \cdot g^\alpha (g^{at})^u = g^{a_\eta} q_{id}^u, L' = L^u, \{K'_j = K_j^u\}_{j \in S})$. Therefore, $tk = (T_w, L', K'_j)$ is a valid search token. $\mathcal{B}$ returns tk to $\mathcal{A}$.

- **Challenge.** The adversary $\mathcal{A}$ sends two equal-length keywords w_0 and w_1 to $\mathcal{B}$. The algorithm $\mathcal{B}$ generates a challenge index as follows:

  1) $\mathcal{B}$ runs H_1-queries twice to obtain $h_0, h_1 \in \mathbb{G}$ such that $H_1(w_0) = h_0$ and $H_1(w_1) = h_1$. For $\eta = \{0,1\}$, let $< w_\eta, h_\eta, a_\eta, c_\eta >$ be the corresponding tuples on the H_1-list. If both $c_0 = 1$ and $c_1 = 1$, then $\mathcal{B}$ reports failure and terminates.
  2) We know that at least one of c_0, c_1 is equal to 0. $\mathcal{B}$ randomly picks $b \in \{0,1\}$ such that $c_b = 0$.
  3) The algorithm $\mathcal{B}$ selects s from $\mathbb{Z}_p$ randomly, and sets $I = (C_{0,1} = g^s, C_{0,2} = h^s)$. Let $\varphi_b = \hat{e}(u_1, u_2)^\gamma \cdot \hat{e}(g, u_2 g^{a_b})^\gamma$. $\mathcal{B}$ runs the above H_2-queries algorithm to obtain $J \in \{0,1\}^{\log^p}$. $\mathcal{B}$ stores the tuple $< \varphi_b, J >$ in the H_2-list and responds to $\mathcal{A}$ with the challenge $I_{w_b} = J$ for a random $J \in \{0,1\}^{log^p}$. Let $\gamma = s$, we have

  $$H_2(\hat{e}(u_1, u_2)^\gamma \cdot \hat{e}(g, H_1(w_b))^\gamma) = J,$$

  i.e $J = H_2(\hat{e}(u_1, u_2)^\gamma \cdot \hat{e}(g, H_1(w_b))^\gamma) = H_2(\hat{e}(u_1, u_2)^\gamma \cdot \hat{e}(g, u_2 g^{a_b})^\gamma)$.

- **Phase 2.** Same as Phase 1. $\mathcal{A}$ can continue to issue the trapdoor queries for keywords w_η, where the only restriction is that $w_\eta \neq w_0, w_1$. $\mathcal{B}$ responds to these queries as before.
- **Guess.** The adversary $\mathcal{A}$ outputs its guess $b' \in \{0,1\}$ of b. $\mathcal{A}$ computes φ_b as follows:

$$\begin{aligned}\varphi_b &= \frac{\hat{e}(C_{0,1}, T_w)}{\hat{e}(L', C_{0,2})} \\ &= \frac{\hat{e}\big(g^s, H_1(w) q_{id}\big)}{(L^u, h^s)} \\ &= \frac{\hat{e}(g^s, g^{a_\eta} g^{atu} g^\alpha)}{\hat{e}\big((g^t)^u, g^{as}\big)} \\ &= \frac{\hat{e}(g,g)^{\alpha s} \cdot \hat{e}(g, g^{a_\eta})^s \cdot \hat{e}(g,g)^{astu}}{\hat{e}(g,g)^{astu}} \\ &= \hat{e}(g,g)^{\alpha s} \cdot \hat{e}(g, g^{a_\eta})^s.\end{aligned}$$

If $\mathcal{A}$ can break our scheme, we have $\varphi_b = \hat{e}(u_1, u_2)^s \cdot \hat{e}(g, g^{a_b})^s$. $\mathcal{B}$ searches I_{w_b} from the H_2-list for φ_b and outputs $\varphi_b / \hat{e}(K_1, u_2 g^{a_b}) = \hat{e}(g,g)^{\alpha\beta\gamma}$. The adversary $\mathcal{A}$ must have issued a query for either $H_2(\hat{e}(u_1, u_2)^\gamma \cdot \hat{e}(g, H_1(w_0))^\gamma)$ or $H_2(\hat{e}(u_1, u_2)^\gamma \cdot \hat{e}(g, H_1(w_1))^\gamma)$. Therefore, with the probability 1/2 the H_2-list contains a pair whose left hand side is $\varphi_\eta = \hat{e}(u_1, u_2)^\gamma \cdot \hat{e}(g, H_1(w_b))^\gamma$. If $\mathcal{B}$ picks this pair (φ_η, I) from the H_2-list, then $\varphi_\eta / \hat{e}(K_1, u_2 g^{a_b}) = \hat{e}(g,g)^{\alpha\beta\gamma}$ as required.

We will analyze that $\mathcal{B}$ correctly outputs $\hat{e}(g,g)^{\alpha\beta\gamma}$ with probability at least ϵ'. The probability that $\mathcal{B}$ does not abort during the simulation phase is at least $1/e$, and the probability that $\mathcal{B}$ does not abort during the challenge phase is at least $1/q_T$. Therefore, $\mathcal{B}$ does not abort with the probability at least $1/eq_T$. In a real attack game $\mathcal{A}$ issues a query for $\varphi_\eta = \hat{e}(u_1, u_2)^\gamma \cdot \hat{e}(g, H_1(w_b))^\gamma$ with probability at least ϵ. The adversary $\mathcal{A}$ issues an H_2 query for either $H_2(\hat{e}(u_1, u_2)^\gamma \cdot \hat{e}(g, H_1(w_0))^\gamma)$ or $H_2(\hat{e}(u_1, u_2)^\gamma \cdot \hat{e}(g, H_1(w_1))^\gamma)$ with probability at least 2ϵ. The detailed analysis of above results is shown in Boneh *et al.*'s scheme [1]. $\mathcal{B}$ will choose the correct pair with probability at least $1/q_{H_2}$. Assuming $\mathcal{B}$ does not abort during the simulation, it will produce the correct answer with probability ϵ/q_{H_2}. Since $\mathcal{B}$ does not abort with the probability at least $1/eq_T$, the probability of $\mathcal{B}$ successfully outputs $\hat{e}(g, g)^{\alpha\beta\gamma}$ with probability $\epsilon/(e \cdot q_T \cdot q_{H_2})$. □

Table 1: Notations

Symbols	Description
P	the pairing operation
E	the group exponentiation in $\mathbb{G}$
E_T	the group exponentiation in $\mathbb{G}_T$
n	the number of attributes in the system
$n_{a,u}$	the number of attributes a client possesses
k	the number of attributes embedded in a ciphertext

5.2 Performance Analysis

In this section, we will give the performance analysis from the perspective of functional comparison, computation cost, and experiment result. In Table 1, we define some notations which will be used in this section.

Functionality comparisons: In Table 2, we give the comprehensive comparisons according to some important features, including expressive, attribute revocation, keyword search and the verifiability. From Table 2, Hur *et al.*'s scheme [7] can achieve fine-grained attribute revocation, but not support data retrieval and result verification. Zheng *et al.*'s scheme [27] can provide verifiability and fine-grained keyword search, but there are huge computational overhead in the verification process. Sun *et al.*'s scheme [19] can achieve data retrieval, the verifiability, and revocation, but the verification progress is low and only support system-level client revocation. Wang *et al.*'s scheme [21] can achieve attribute revocation and keyword search, but the verifiability of search results is not considered. In general, compared with the above schemes, our scheme has better functionality.

Computation cost: In Table 3, since we have the same functionality as Sun *et al.*'s scheme [19], we briefly compare our computational costs with Sun *et al.*'s. As the operation cost over $\mathbb{Z}_p$ is much less than group and pairing operation, we ignore the computational time over $\mathbb{Z}_p$. From Table 3, In *Setup* algorithm, Sun *et al.*'s scheme needs $3n$ exponentiations in $\mathbb{G}$, one exponentiation in $\mathbb{G}_T$, and one pairing operation, while our scheme only requires one exponentiations in $\mathbb{G}$, one exponentiation in $\mathbb{G}_T$, and one pairing operation. In *Keygen* algorithm, our scheme needs $(3+n_{a,u})$ exponentiations in $\mathbb{G}$, but Sun *et al.*'s scheme needs $(2n+1)$ exponentiations in $\mathbb{G}$ and two exponentiations in $\mathbb{G}_T$. In *Encrypt* algorithm, our scheme needs $(3k+2)$ exponentiations in $\mathbb{G}$, three exponentiations in $\mathbb{G}_T$ and one pairing operation, but Sun *et al.*'s scheme needs $(n+1)$ exponentiations in $\mathbb{G}$, one exponentiation in $\mathbb{G}_T$ and one pairing operation. The time cost of our scheme is a little larger than Sun *et al.*'s scheme. In *Trapdoor* algorithm, our scheme needs $(k+4)$ exponentiations in $\mathbb{G}$, However, Sun *et al.*'s scheme needs $(2n+1)$ exponentiations in $\mathbb{G}$, which is larger than our scheme. In *Search* algorithm, Sun *et al.*'s scheme requires one exponentiation and $(n+1)$ pairing operations, but our scheme only needs two exponentiations in $\mathbb{G}_T$ and two pairing operations.

Experiment result: We conduct our experiments using Java Pairing-Based Cryptography (JPBC) library [2]. We implement the proposed scheme on a Windows machine with Intel Core 2 processor running at 3.30 GHz and 4.00 G memory. The running environment of our scheme is Java Runtime Environment 1.7, and the Java Virtual Machine(JVM) used to compile our programming is 64 bit which brings into correspondence with our operation system.

In our experiments, we compare the proposed scheme with Sun *et al.*'s scheme [19] and Wang *et al.*'s scheme [21] in the search time. The modulus of the elements in the group is chosen to be 512 bits, the number of attributes ranges from 10 to 50.

From Figure 3, we know that the search time cost grows linearly with the number of attributes in Sun *et al.*'s scheme, and the search time cost of Wang *et al.*'s scheme is less than Sun *et al.*'s scheme. However the search time cost of the proposed scheme is the most efficient than the other two schemes. For example, when the number of attributes is 50, the search time consumption of the proposed scheme only needs 0.03s, while Sun *et al.*'s scheme needs about 0.8s and Wang *et al.*'s scheme needs 0.05s. Therefore, compared with the above two schemes, the proposed scheme is more efficient and practical.

6 Conclusions

In this paper, we have proposed a verifiable attribute-based keyword search encryption with attribute revocation for electronic health record system. By using a KEK tree and re-encryption techniques, the proposed scheme can achieve efficient revocation and assure that the updated ciphertext cannot be decrypted by the revoked

Table 2: The comparisons of the functionality

Schemes	Expressive	Revocation	Keyword Search	Verifiability
Hur et al.*'s scheme [7]*	access tree	√	×	×
Sun et al.*'s scheme [19]*	AND gate	√	√	√
Wang et al.*'s scheme [21]*	LSSS	√	√	×
Zheng et al.*'s scheme [27]*	access tree	×	√	√
Ours	LSSS	√	√	√

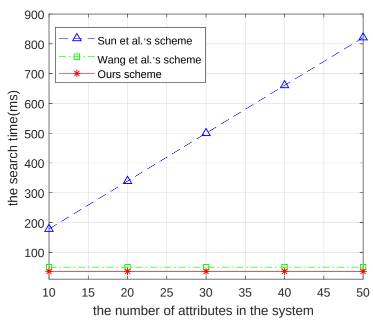

Figure 3: The time cost of EHR system

clients. In addition, we introduce TPA to verify the integrity of the returned search results, which can reduce the client's computation overhead. Furthermore, performance analysis shows that our scheme is efficient and practical for electronic health record system. Since the policy may contain some sensitive information of patient. The proposed scheme do not support policy hiding. For the future work, we intend to propose a privacy-preserving attribute-based keyword search encryption scheme.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grants No. 61807026, the Natural Science Basic Research Plan in Shaanxi Province of China under Grant No. 2019JM-198, the Plan For Scientific Innovation Talent of Henan Province under Grant No. 184100510012, and in part by the Program for Science and Technology Innovation Talents in the Universities of Henan Province under Grant No. 18HASTIT022.

Table 3: The comparisons of computation cost

Operations	Sun *et al.*'s scheme [19]	Ours
Setup	$3nE + E_T + P$	$E_T + E + P$
KeyGen	$(2n+1)E + 2E_T$	$(3 + n_{a,u})E$
Encrypt	$(n+1)E + E_T + P$	$(3k+2)E + 3E_T + P$
Trapdoor	$(2n+1)E$	$(4+k)E$
Search	$(n+1)P + E_T$	$2P + 2E_T$

References

[1] D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in *International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 506–522, May 2004.

[2] A. D. Caro and V. Iovino, "jPBC: Java pairing based cryptography," in *Proceedings of The 16th IEEE Symposium on Computers and Communications (ISCC'11)*, pp. 850–855, 2011.

[3] P. S. Chung, C. W. Liu, and M. S. Hwang, "A study of attribute-based proxy re-encryption scheme in cloud environments," *International Journal of Network Security*, vol. 16, no. 1, pp. 1–13, 2014.

[4] M. L. Florence and D. Suresh, "Enhanced secure sharing of PHRs in cloud using attribute-based encryption and signature with keyword search," *Advances in Big Data and Cloud Computing*, vol. 645, pp. 375–384, 2018.

[5] R. Gandikota, "A secure cloud framework to share EHRs using modified CP-ABE and the attribute bloom filter," *Education and Information Technologies*, vol. 23, no. 5, pp. 2213–2233, 2018.

[6] M. T. Hu, H. Gao, and T. G. Gao, "Secure and efficient ranked keyword search over outsourced cloud data by chaos based arithmetic coding and confusion," *International Journal of Network Security*, vol. 21, no. 1, pp. 105–114, 2019.

[7] J. Hur and D. K. Noh, "Attribute-based access control with efficient revocation in data outsourcing systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 7, pp. 1214–1221, 2011.

[8] M. Joshi, K. Joshi, and T. Finin, "Attribute based encryption for secure access to cloud based EHR systems," in *Proceedings of IEEE 11th Intenational Conference on Cloud Computing*, pp. 932–935, July 2018.

[9] M. Li, S. C. Yu, Y. Zheng, K. Ren, and W. J. Lou, "Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 1, pp. 131–143, 2013.

[10] Y. Miao, J. Ma, X. Liu, and F. Wei Z. Liu, L. Shen, "VMKDO: Verifiable multi-keyword search over encrypted cloud data for dynamic data-owner," *Peer-to-Peer Networking and Applications*, vol. 11, no. 2, pp. 287–297, 2018.

[11] D. Naor, M. Naor, and J. Lotspiech, "Revocation and tracing schemes for stateless receivers," in *Proceedings of The 21st Annual International Cryptology Conference*, pp. 41–62, Aug. 2001.

[12] S. Narayan, M. Gagne, and R. S. Naini, "Privacy preserving HER system using attribute-based infrastructure," in *Proceedings of The ACM Workshop on Cloud Computing Security Workshop*, pp. 47–52, Oct. 2010.

[13] Y. S. Rao, "A secure and efficient ciphertext-policy attribute-based signcryption for personal health records sharing in cloud computing," *Future Generation Computer System*, vol. 67, pp. 133–151, 2017.

[14] B. E. Reedy and G. Ramu, "A secure framework for ensuring EHR's integrity using fine-grained auditing and CP-ABE," in *Proceedings of The IEEE 2nd Intenational conference on Big Data Security*, pp. 85–89, Apr. 2016.

[15] S. Rezaei, M. A. Doostari, and M. Bayat, "A lightweight and efficient data sharing scheme for cloud computing," *International Journal of Electronics and Information Engineering*, vol. 9, no. 2, pp. 115–131, 2018.

[16] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Proceedings of The 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 457–473, May 2005.

[17] D. Sethia, H. Saran, and D. Gupta, "CP-ABE for selective access with scalable revocation: A case study for mobile-based healthfolder," *International Journal of Network Security*, vol. 20, no. 4, pp. 689–701, 2018.

[18] D. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *Proceedings of IEEE Symposium on Security and Privacy*, pp. 44–55, May 2000.

[19] W. Sun, S. Yu, W. Lou, Y. Hou, and H. Li, "Protecting your right: Verifiable attribute-based keyword search with fine-grained owner-enforced search authorization in the cloud," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 4, pp. 1187–1198, 2016.

[20] S. F. Tzeng, C. C. Lee, and M. S. Hwang, "A batch verification for multiple proxy signature," *Parallel Processing Letters*, vol. 21, no. 1, pp. 77–84, 2011.

[21] S. P. Wang, D. Zhang, Y. Zhang, and L. Liu, "Efficiently revocable and searchable attribute-based encryption scheme for mobile cloud storage," *IEEE Access*, vol. 6, pp. 30444–30457, 2018.

[22] B. Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization," in *Proceedings of The 14th International Conference on Practice and Theory in Public Key Cryptography (PKC'11)*, pp. 53–70, Mar. 2011.

[23] A. Wu, D. Zheng, Y. L. Zhang, and M. L. Yang, "Hidden policy attribute-based data sharing with direct revocation and keyword search in cloud computing," *Sensors*, vol. 18, no. 7, pp. 1–17, 2018.

[24] F. Xhafa, J. F. Wang, X. F. Chen, J. K. Liu, J. Li, P. Krause, and D. S. Wong, "An efficient PHR service system supporting fuzzy keyword search and fine-grained access control," *Soft Computing*, vol. 18, no. 9, pp. 1795–1802, 2014.

[25] S. Yu, C. Wang, K. Ren, and W. Lou, "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, Apr. 2010.

[26] Y. Zhao, M. Ren, S. Jiang, G. Zhu, and H. Xiong, "An efficient and revocable storage CP-ABE scheme in the cloud computing," *Computing*, pp. 1–25, 2018.

[27] Q. Zheng, S. Xu, and G. Ateniese, "VABKS: Verifiable attribute-based keyword search over outsourced encrypted data," in *Proceedings of IEEE Conference on Computer Communications*, pp. 522–530, Apr. 2014.

Biography

Zhenhua Liu received the B. S. degree from Henan Normal University, and the master's and Ph.D. degrees from Xidian University, China, in 2000, 2003, and 2009, respectively. He is currently a Professor with Xidian University. His research interests include cryptography and information security.

Yan Liu received the B. S. degree from Shenyang Agricultural University in 2017. She is currently pursuing the master's degree in mathematics with Xidian University. Her research interests include cryptography and cloud security.

Jing Xu received the B. S. degree from Henan Normal University in 2017. She is currently pursuing the master's degree in mathematics with Xidian University. Her research interests include cryptography and cloud security.

Baocang Wang received the B. S. degree in Computational Mathematics and Their Application Softwares from Xidian University, China, in 2001, the M. S. degree in Cryptology from Xidian University, China, in 2004, and the Ph.D. degree in Cryptology from Xidian University, China, in 2006. He is currently a Professor and Ph. D. supervisor with Xidian University. His research interests include postquantum cryptography, fully homomorphic cryptography, number theoretic algorithms, and cloud security.