A revocable attribute-based encryption EHR sharing scheme with multiple authorities in blockchain

Xiaohui Yang1 · Wenjie Li1,2 · Kai Fan2

Received: 15 September 2021 / Accepted: 8 September 2022 / Published online: 23 September 2022
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2022

Abstract
With the development of digital healthcare, sharing electronic medical record data has become an indispensable part of improving medical conditions. Aiming at the centralized power caused by the single attribute authority in current CP-ABE schemes and the problem that cloud servers are curious and even malicious, we design a revocable CP-ABE EHR sharing scheme with multiple authorities (MA-RABE) in blockchain. In this solution, a group of authorities complete user attribute distribution, key generation and user management through secret sharing and transactions. Besides, we innovatively implemented a distributed one-way anonymous key agreement so that other participants cannot obtain useful information from the fully hidden policy embedded in the ciphertext. Taking into account the computational overhead of a large number of bilinear operations in the decryption process, the solution also supports the cloud server to pre-decrypt the ciphertext, and the data user only needs to perform exponentiation operation once to obtain the plaintext from the pre-decryption result. Theoretical analysis and performance evaluation show that the scheme has reliable security and lower user revocation and ciphertext update overhead.

Keywords Blockchain · Data sharing · Attribute-based encryption · EHR sharing

1 Introduction

To better trace the health conditions of patients, Electronic Health Record (EHR) emerged, which greatly improved the level of public healthcare management [1]. Massive studies have shown that analyzing and studying a large amount of timely EHR data can respond to public health emergencies such as the current COVID-19 pandemic [2] and provide a basis for public health decision-making.

When patients go to the hospital, they need to share their health records and medical histories with the doctor. Besides, the patients can contribute their medical data to medical research institutions under their will. Therefore, sharing medical data is an essential step to improve the quality of healthcare providers and make the healthcare system smarter [3]. However, the EHR management systems maintained independently by healthcare providers have led to the lack of interoperability among stakeholders [4], causing a waste of medical resources during hierarchical referrals, so that patients undoubtedly lose control of their own EHR. Moreover, this EHR management model lacks transparency and is vulnerable to single points of failure, internal leakage and phishing attacks [5].

In addition, the security and privacy of EHR data have also attracted widespread attention and research in academia. Security here mainly refers to confidentiality, because EHR contains a large amount of critical and highly private patient information, including physiological information, medical history and even personal information [6]. Therefore, EHR must not be stored in the third party in plaintext, and only EHR owners and authorized users can access EHR. To achieve privacy protection, even if the EHR is stored in ciphertext, the relevant access policy must not disclose the personal information of participants during the sharing.

1 School of Cyber Security and Computer, Hebei University, Baoding, Hebei 071000, China
2 State Key Laboratory of Integrated Service Networks, Xidian University, Xi’an, Shanxi 710071, China
Aiming at the security, privacy and interoperability of EHR data, the traditional symmetric encryption technology can protect the confidentiality of EHR data, but it is difficult to meet the requirement for flexibility of EHR data sharing [7]. The novel attribute-based encryption (ABE) technology has better flexibility and fine-graininess and has become a promising solution for sharing EHR [8].

Technically speaking, ABE can be divided into two types: ciphertext policy attribute-based encryption (CP-ABE) and key policy attribute-based encryption (KP-ABE) [9]. KP-ABE embeds the access policy into the decryption key of Data User(DU) and embeds a set of attributes into the ciphertext. In CP-ABE, DU’s decryption key corresponds to a set of attributes, and the ciphertext in the cloud server is associated with an access policy. Only DU, whose attributes satisfy the access structure, can decrypt the ciphertext associated with the access structure. Both ABE schemes implement fine-grained access control through access structure customization and attribute management [10]. However, KP-ABE is more suitable for static access scenarios, such as paid streaming video websites and log encryption management. These scenarios hardly change the attributes associated with ciphertext. In dynamic access scenarios, especially EHR sharing scenarios, the attributes of EHR accessors are generally stable, and the EHR owner needs to change access policies embedded in ciphertext according to the access scenarios. Obviously, CP-ABE can better solve the interoperability problem between stakeholders since it allows Data Owner (DO) to customize a fine-grained access structure for data to decide who can access them. Therefore, CP-ABE is considered to be an ideal solution for securely sharing EHR in the public cloud [11].

However, there are some problems existing in the original CP-ABE [12, 13], the access policy embedded in the ciphertext is publicly accessible. If the data sharing scheme directly utilizes this type of explicit access policy, then the sensitive personal information of DU and DO will be indirectly derived [7]. Moreover, CP-ABE uses massive bilinear calculations in the encryption and decryption process, which consumes expensive time and storage resources and limits the application scenarios of CP-ABE [14, 15].

Besides, in most CP-ABE schemes, a single Attribute Authority (AA) manages the distribution and revocation of all attributes. Such schemes are vulnerable to a single point of failure and further affect the availability of attribute management. Due to the centralized power, the malicious authority even abuses the private key and distributes the attribute key to illegal users, resulting in unauthorized access to data [16, 17]. Although there have been some multi-AA schemes where the attribute universe is divided into multiple management domains, and each AA manages a disjoint attribute set. However, it causes the trust problem between management domains and does not fundamentally solve the single point of failure [18].

Fortunately, the emerging blockchain technology can ensure the integrity and immutability of data, so that operations on the blockchain can be recorded openly and transparently in the form of transactions. Therefore, the blockchain is expected to solve the trust crisis among the stakeholders and effectively improve the interaction and collaboration of the healthcare industry [3]. We can utilize the blockchain to avoid the problem of centralized power in CP-ABE and provide a new paradigm for health information exchange (HIE) [19].

To solve the above challenges faced by CP-ABE, avoid the centralized power caused by single AA, and ensure the reliable authorization of users and the secure sharing of data, we propose a secure Revocable ABE scheme with Multiple Authorities (MA-RABE) in blockchain. Specifically, the main contributions of our proposed scheme MA-RABE are summarized as follows:

1. **Decentralized power.** To prevent the abuse of private keys and single points of failure caused by a single AA, we deploy the entire scheme in blockchain. A group of attribute authorities (hereinafter referred to as AAs) through secret sharing jointly collaborate to generate global parameters, distribute keys and manage users.

2. **Fully Hidden policy.** The rows of the share-generating matrix are replaced by implicit bilinear mappings about the attributes. Only the attribute recover key owner can calculate which rows of the matrix the attribute corresponds to through the one-way anonymous key agreement. And AAs complete the secret distribution process of the attribute obfuscation key under the decentralized setting.

3. **Efficient user management and key update.** The AAs manage the legal users through the user binary tree, and the concrete operations include registration, revocation and rejoining. After the legal user set is changed, the AAs will generate the key update message for the minimum covered node set for the legal user to achieve a low key update overhead. Thus only the user with a node in minimum covered node set on the path to the root node can generate the pre-decryption key.

4. **Pre-decryption of ciphertext.** CSP can calculate the pre-decryption key against the node key and KU stored in the blockchain, and pre-decrypt the ciphertext to obtain the intermediate ciphertext. The DU only needs to perform one exponentiation on the intermediate ciphertext to obtain the plaintext.

The rest of the paper is organized as follows. We introduce the work related to the EHR sharing scheme in Sect. 2. We explain some preliminaries about encryption primitives
and the technologies used in Sect. 3. Next, we describe the program model, scheme definition and security model in Sect. 4. Section 5 talks about the scheme construction in detail. Security analysis and performance evaluation of the scheme are presented in Sects. 6 and 7 respectively. Finally, we summarize the proposed scheme in Sect. 8.

2 Related work

CP-ABE is a promising data sharing solution. With the increased requirements in data privacy and application scenario variety, CP-ABE has also been improved to extend more functions, such as user revocation and computation outsourcing. The related work is as follows.

2.1 Multiple attribute authorities

The construction of the original multi-AA scheme [12] is still based on a trusted central authority (CA) and a global identifier (GID). The CA in the scheme can even decrypt each ciphertext. Chase and Chow [13] delete the trusted CA and achieve privacy protection by preventing CA from collecting information from specific users. Li et al. [17] implemented a multi-AA CP-ABE scheme TMACS for public cloud storage by using \((t,n)\) threshold secret sharing. Any \(t\) AAs can generate the secret key for the legitimate user. And as long as less than \(t\) AAs are compromised, TMACS can guarantee security. Unfortunately, the scheme did not implement user revocation and outsourcing computation. Zhong et al. [20] proposed a decentralized multi-authority CP-ABE access control scheme. Each AA can independently publish attributes in its management domain. It realizes user revocation by only distributing version keys to legitimate users. The obfuscated access policy is designed in LSSS to protect the policy privacy. But due to each AA still manages disjoint attribute sets independently, it still cannot overcome the single-point bottleneck. Similarly, Sarma et al. [21] proposed a multi-authority scheme MACFI, in which each AA is an independent organization. Each AA manages a set of disjoint attributes, and the AA assigns the corresponding attributes to the user after verifying the user’s role. To reduce the communication overhead on the user side, Qin et al. [22] proposed a blockchain-based access control scheme BMAC, which leverages consortium blockchain to establish trust between multiple AAs. BMAC also introduces Shamir secret sharing to achieve cross-domain management of attribute tokens. Each attribute is jointly managed by multiple authorities, thus avoiding a single point of failure.

2.2 Policy privacy

Given the privacy leakage caused by partially hiding the attribute value in the access policy, Ramu [23] proposed a secure cloud framework by combining a modified CP-ABE and attribute bloom filter (ABF), which realizes the attribute hidden entirely. Similarly, Hao et al. [7] designed a fuzzy attribute positioning mechanism based on the garbled bloom filter to prevent unauthorized users from obtaining valuable attribute information. To avoid the abuse of the private key, Wu et al. [24] utilized blockchain technology to ensure the integrity and non-repudiation of the data and proposed an ABF-based and privacy-preserving traceable CP-ABE scheme. Fan et al. [25] realized the anonymization of attributes by adopting the one-way anonymous key agreement [26], where attributes are verifiable and anonymous. Besides, verifiable outsourcing decryption in edge is introduced to reduce the computing overhead on the user side. By exploiting hidden vector encryption, Zhang et al. [27] achieved complete policy hiding, where DU obtains the minimum authorized sets of \((M, \rho)\) instead of the entire \((M, \rho)\). Yang et al. [28] designed an authorization method based on the Private Set Intersection, which can completely hide the policy by calculating the intersection of the user’s attribute set and the policy’s attribute set.

2.3 User revocation

To strengthen the system security and meet the application requirements in resource-constrained scenarios, Wei et al. [29] proposed a revocable hierarchical scheme RS-HABE, which expanded user revocation, secret key delegation and ciphertext update to the original ABE. Xiong et al. [30] designed a traceable CP-ABE scheme that employed outsourcing decryption to minimize the computing overhead on the user side. When encrypting data, the data owner needs to embed the user ID set and access policy into ciphertext. Only DU satisfies both two restrictions, the ciphertext can be decrypted, which yet runs counter to fuzzy encryption. Zheng et al. [31] proposed a cloud-assisted user revocation data sharing scheme to protect IoT data. It maintains effective sharing and user revocation but does not take into account the privacy leakage caused by unprocessed policies. Liu et al. [32] proposed a searchable ABE scheme BC-SABE that supports user revocation and outsourcing decryption, in which the blockchain replaces the centralized server. A group of AAs completes the generation of public parameters, key management and user revocation on the blockchain. However, this solution also has the problem of privacy leakage of unprocessed policies. Besides, Guo et al. [33] designed a user revocation mechanism using the chameleon hash function. To resist the collusion attack between the revoked users and the malicious users, Zhang
et al. [34] set a group manager to assign a certificate for each user and embed the certificate into the user’s secret key. While implementing revocable user management, the scheme TR-TABE [35] can achieve white-box traceability without maintaining a user list.

In general, the above solutions cannot take into account full policy hiding, thorough decentralization, and efficient outsourcing of pre-decryption. However, our proposed MA-RABE achieves the above functions perfectly. In addition, MA-RABE can implement a complete user management process, including user revocation and rejoining.

## 3 Preliminaries

In this section, we give a review of some preliminaries to better understand the proposed scheme.

### 3.1 Bilinear maps

**Definition 1** Let $\mathcal{G}$ and $\mathcal{G}_t$ be two multiplicative cyclic groups of prime order $p$, where $g$ is a generator of $\mathcal{G}$. Let $\hat{e}$ be a bilinear map, $\hat{e} : \mathcal{G} \times \mathcal{G} \rightarrow \mathcal{G}_t$. And $\hat{e}$ has following three properties:

- **Bilinearity:** $\forall a, b \in \mathbb{Z}_p$ and $\forall u, v \in \mathcal{G}$, we have $\hat{e}(g^a, g^b) = \hat{e}(g, g)^{ab}$.
- **Computability:** $\forall u, v \in \mathcal{G}$, $\hat{e}(u, v)$ can be computed in a polynomial time with an efficient algorithm.
- **Non-degeneracy:** $\hat{e}(g, g) \neq 1_{\mathcal{G}_t}$, where $1_{\mathcal{G}_t}$ is the identity of $\mathcal{G}_t$.

Besides, the bilinear map $\hat{e}$ is considered symmetric because it satisfies $\hat{e}(g^a, g^b) = \hat{e}(g, g)^{ab} = \hat{e}(g^b, g^a)$.

### 3.2 Access structure

**Definition 2** Let $\{A_1, A_2, ..., A_n\}$ be the attribute universe. A collection $\mathcal{A} \subseteq 2^{\{A_1, A_2, ..., A_n\}}$ is monotone if for any two attribute sets $B$ and $C$ : if $B \in \mathcal{A}$ and $B \subseteq C$, then $C \in \mathcal{A}$. An access structure (respectively, monotone access structure) is a collection (respectively, monotone collection) $\mathcal{A}$ of non-empty subsets of $\{A_1, A_2, ..., A_n\}$, i.e. $\mathcal{A} \subseteq 2^{\{A_1, A_2, ..., A_n\}} \setminus \emptyset$. The sets in $\mathcal{A}$ are referred as the authorized sets, and the sets not in $\mathcal{A}$ are called the unauthorized sets.

### 3.3 Linear secret sharing scheme

Let $p$ be a prime. There is a matrix $M$ with $l$ rows and $m$ columns. The linear secret sharing scheme $\Pi$ over the attribute Universe $\mathcal{A}$ is considered linear if:

- the shares of a secret $s$ composes a vector over $\mathbb{Z}_p^l$.
- For $\Pi$, there is a function $\rho$ mapping from $\{1, 2, ..., l\}$ to $\mathcal{A}$. Therefore the $i^{th}$ of $M$ will be associated with $\rho(i)$. Given a column vector $\tilde{v} = (s, r_2, ..., r_m)^T$, where $s \in \mathbb{Z}_p$ is the secret to be shared and $r_2, r_3, ..., r_m$ are randomly selected from $\mathbb{Z}_p$. $M\tilde{v}$ are $l$ shares of $s$ based on $\Pi$, so the $i^{th}$ share $\lambda_i = (M\tilde{v})_i$ belongs to the attribute $\rho(i)$.

To reconstruct the secret $s$, the user whose attribute set $S$ satisfy the access policy $(M, \rho)$ can find a set of constants $(\alpha_1, \alpha_2, ..., \alpha_l)$ in a polynomial time so that $\sum_{i \in I} w_i M_i = (1, 0, ..., 0)$, where $I$ denotes the rows which are associated with attributes in $S$. The the secret $s$ can be recovered by the following formula:

$$
\sum_{i \in I} w_i \lambda_i = \sum_{i \in I} w_i (M\tilde{v})_i = \sum_{i \in I} (w_i M_i)\tilde{v} = (1, 0, ..., 0)(s, r_2, ..., r_m)^T = s
$$

### 3.4 Pedersen secret sharing scheme

Pedersen secret sharing scheme [36], improved from Shamir’s secret sharing scheme [37], is also a kind of $(k, n)$ threshold secret sharing scheme. Shamir’s secret sharing scheme requires a trusted third party to divide the master secret into $n$ sub-secrets through a $k - 1$ degree polynomial and distribute them to $n$ participants $P = \{P_1, P_2, ..., P_n\}$. Reconstruction of the master secret requires at least $k$ participants. The advantage of Pedersen’s scheme is that it enables $n$ participants to divide the master secret without relying on a third-party CA. The Pedersen secret sharing scheme algorithm is described as follows:

- **master secret generation:** For each of the $n$ participants, to generate a $k - 1$ degree polynomial $f_i(x)$, $P_i$ independently and randomly select $k$ coefficients $a_{i,c} \in \mathbb{Z}_p^l (c = 0, ..., k - 1)$. Among them, the constant term coefficient $a_{i,0}$ is used as its own sub-secret $a_i$. The master secret for the entire scheme is $s = \sum_{i=1}^n a_i$, in which $s$ is implicit.
- **master share computation:** Then the participant $P_i$ generates $n$ sub-shares $s_{ij} = f_i(j) (j = 1, ..., n)$ and sends $s_{ij}$ to $P_j$ through a secure channel. After $P_j$ receives $n - 1$ sub-shares from $P_i$, $P_j$ calculates its master share $s_j = \sum_{i=1}^n s_{ij}$.
- **master secret reconstruction:** To recover the master secret, at least $k$ participants are required. Suppose the set of these participants is $P_{rec} (k \leq P_{rec} \leq n)$, and the master secret $s$ can be reconstructed by using LaGrange interpolation as follow:
\[ s = \sum_{i \in P_{mc}} s_i \prod_{j \in P_{mc}, j \neq i} \frac{j}{j-1} \]  

(2)

For the convenience of subsequent calculations, we set 
\[ L(i) = \prod_{j \in P_{mc}, j \neq i} \frac{j}{j-1} \]

### 3.5 Blockchain

Blockchain is a decentralized distributed ledger technology. There are two types of nodes in the blockchain network: full nodes and light nodes. Full nodes (also known as miner nodes) elect one to pack the next block through the consensus protocol. The elected miner node packs transactions into a block and broadcasts it to other nodes [38]. After verifying the validity of the newly generated block, the full nodes will append the received block into the chain structure locally. Therefore, full nodes are required to have higher storage and computing capabilities. On the other hand, light nodes do not participate in the generation of blocks, but they can verify the legitimacy of blocks/transactions with a simple calculation.

All nodes that join the blockchain network can access the data in blockchain. Similarly, they can also call the functions in smart contracts deployed in blockchain. With the help of smart contracts, data sharing intermediaries can provide services to stakeholders in the form of DAPPs [39, 40]. Related operations such as attribute distribution and user management will become open and transparent [41].

### 3.6 User binary tree

In some papers, User Binary Tree (BT) [42, 43] is also called KEY Tree, in which each user is assigned a unique leaf node. If we calculate the update keys for each non-revoked user in the form of a list, the complexity is \( O(N - R) \), where \( N \) is the number of users and \( R \) is the number of revoked users. In BT, the update keys only need to be calculated for the minimum covered nodes, the complexity is \( O(N \log N/R) \) [32].

The KUNode algorithm is as detailed in Algorithm 1, where \( BT \) is a user binary tree, \( revl \) is the user revocation list, \( rejl \) is the user rejoin list, and \( x_L \) and \( x_R \) are the left and right child nodes of the node \( x \) respectively. Note that non-revoked users in the scheme refer to users who have never been revoked or users who have been rejoined after revoked. The users who have not been rejoined after revoked are called illegal users.

To further describe the KUNode algorithm in MABRABE, a user binary tree with a depth of 4 and 8 users marked A-H is shown in Fig. 1. The users crossed out by the red line represent illegal users. At the end of the KUNode algorithm, the nodes in set \( X \) are marked orange, and the nodes in set \( Y \) are marked green. When all users are non-revoked users, the set \( Y \) returned by the KUNode algorithm only contains the root node, so the key update needs to be generated for the root node. When users \( C, G \) and \( H \) are illegal, the key update needs to be generated for nodes 4, 6 and 11.

![Fig. 1](image_url)  

The nodes need to be generated the key update

#### Algorithm 1 KUNode Algorithm

Input: \( revl, rejl, BT, t \)  
Output: the minimum covered nodes

1: \( X, Y \leftarrow \emptyset \)
2: for each \((v_i, t_i) \in A\) do
3: \( v_t = \text{revl}[v_i] \)
4: \( t_t = \text{time} - t_i \)
5: \( \text{if} \ v_t > \text{time} \) then \( X \leftarrow X \cup \text{Path}(v_i) \)
6: \( \text{end if} \)
7: \( \text{end for} \)
8: for each \( x \in X \) do
9: \( v_x = \text{Y} \)
10: \( t_x = \text{time} \)
11: \( \text{if} \ v_x > \text{time} \) then \( Y \leftarrow Y \cup x \)
12: \( \text{end if} \)
13: \( \text{end for} \)
14: \( \text{if} \ X = \emptyset \) then \( Y \leftarrow \text{root} \)
15: \( \text{end if} \)
16: \( \text{return} Y \)

### 3.7 The complexity assumption

Let \( G \) and \( G_j \) be two multiplicative cyclic groups of prime order \( p \), where \( g \) is a generator of \( G \). Let \( \bar{e} \) be a bilinear map, \( \bar{e} : G \times G \rightarrow G_1 \). Given four random numbers \( a, b, c, z \in \mathbb{Z}_p^* \), three group elements \( g^a, g^b, g^c \in G \) and one random group element \( Z \in G_j \). The problem of determining if \( Z = \bar{e}(g, g)^{abc} \) or \( Z = \bar{e}(g, g)^z \) is called the decisional Bilinear Diffie-Hellman (DBDH) problem.

**Definition 3** It is said that DBDH assumption holds on bilinear group \((G, G_1, \bar{e}, p)\) if there does not exist a probabilistic polynomial time (PPT) adversary \( A \) who can solve the above the DBDH problem with a non-negligible advantage \( Pr_{A, DBDH} \).

### 4 System overview

This section provides an overview of the proposed model and the security requirements to satisfy. Besides, we introduce the main functions in the scheme and define the security model.
4.1 System model

As shown in Fig. 2, there are four kinds of entities in MA-RABE:

Data Owner (DO) The DO has a large amount of EHR data. The DO encrypts the EHR by customizing the LSSS policy, then outsources the ciphertext with the corresponding access policy to the CSP.

Data User (DU) The DU refers to the research institution and the medical provider, who needs to register its key and attributes in AAs. To access data of DO, DU can initiate a pre-decryption request to CSP, then obtain the symmetric key by performing one exponentiation on the returned pre-decryption result.

Attribute Authority (AA) AAs manage all users based on a global $BT$, $revl$ and $rejl$. AAs are responsible for distributing Attribute Recovery KEY (ARK), Node Key (NK) and Update Key (UK) for non-revoked DU through PSSS.

Cloud Service Provider (CSP) The CSP is responsible for storing the ciphertext uploaded by the DO and providing pre-decryption services to all DUs. Note that we assume that CSP is semi-trusted, that is, CSP is honest but curious. Therefore, CSP will actively leak the privacy of the policy embedded in the ciphertext, but it will execute the protocol honestly.

Besides, due to the trust problem of multiple parties involved in a distributed environment, the blockchain is used as the underlying architecture of the solution. The blockchain contains the system smart contract of this solution (hereinafter referred to as the contract). The contract provides AAs with the registration function, the revocation function and the rejoining function. The contract provides interfaces for AAs to call the PSSS protocol, which are used to negotiate and generate keys. The contract also contains a user binary tree and two lists $revl$ and $rejl$ that can store $N$ tuples.

4.2 Security requirements

Robustness The system can tolerate some malicious attribute authorities, and these malicious attribute authorities will not affect the normal operation of the system and will not control the system.

Decentralization The operation of the system is not under the control of a central entity. Especially in terms of user management and key distribution. In addition, the solution should not be threatened by a single point of failure.

Collusion resistance Even if the attribute set of colluders satisfies the access policy, they can not decrypt the symmetric key encrypted by LSSS, so they will not get the plaintext further.
Policy privacy Only the policy constitutors and the attribute owners can reconstruct the mappings from attributes to the rows in the share-generating matrix. In addition, after receiving the pre-decryption request information from the DU, the CSP cannot deduce which attributes the DU has.

4.3 Scheme definition

In this section, we provide the formal definition of the scheme MA-RABE.

- **Setup** ($\lambda, N$) $\rightarrow (GP, BT, revl, rejf)$. AAs execute this function, which takes one random security parameter $\lambda$ and the maximum number of users $N$ as input. Then the function outputs the global parameter GP, the initialized DU binary tree $BT$ and two empty lists $revl$ and $rejf$, which store the user’s revocation and rejoining time respectively.

- **NKGen** ($GP, uid, S_{uid}, BT$) $\rightarrow NK_{uid}$. This function takes the public parameters GP, user identifier $uid$, user’s attribute set $S_{uid}$ and $BT$ as input and outputs the user’s node key $NK_{uid}$. The node key is a set of calculations of the user public key and attributes in cyclic group, so it can be regarded as the issuance for user attributes.

- **UKGen** ($GP, revl, rejf, BT, t$) $\rightarrow UK_t$. The function takes the public parameters GP, the revocation list $revl$, rejoins the list $rejf$, $BT$ and time $t$ as input, and calculates the update key $UK_t$ for the non-revoked user at the time $t$.

- **PDKGen** ($GP, BT, NK_{uid}, UK_t$) $\rightarrow PDK_{\theta, uid}$. The function takes public parameters GP, user node key $NK_{uid}$, update key $UK_t$ and $BT$ as input. Take the intersection node $\theta$ to compute and output the user’s pre-decryption key $PDK_{\theta, uid}$, and return False if there is no intersection node.

- **Encrypt** ($MSG, (M, \rho), GP, t$) $\rightarrow CT$. The function takes plaintext $MSG$, a monotonic access structure $(M, \rho)$, public parameters $GP$ and time $t$ as input. Then it outputs ciphertext $CT$.

- **PreDec** ($CT, PDK_{\theta, uid}, NK_{uid}, UK_t$) $\rightarrow CT'$. This function inputs the ciphertext $CT$, the user’s Pre-Decryption Key $PDK_{\theta, uid}$, node key $NK_{uid}$ and update key $UK_t$, then outputs the pre-decryption result $CT'$.

- **UserDec** ($CT', sk_{uid}$) $\rightarrow MSG$. This function inputs the pre-decryption result $CT'$ and the user private key $sk_{uid}$, and outputs the plaintext $MSG$.

4.4 Security model

The Indistinguishability security under Chosen-Plaintext Attacks (IND-CPA security) of the MA-RABE is defined by the following game between a PPT adversary $A$ and the challenger $C$. Assuming that $\mathcal{A}$ is the set of AAs with the size of $n$ and the scheme satisfies the security of $(k, n)$ PSSS, that is, there are at most $k - 1$ compromised AAs in $\mathcal{A}$.

**Setup** The adversary $A$ specifies a corrupted authority set $\mathcal{A}'$ with the size of $n' \leq k - 1$. The challenger $C$ runs the Setup() to generate the global parameters GP, and the binary tree $BT$, as well as two lists $revl$ and $rejf$ that store user’s revocation and rejoining time respectively.

**Query phase 1** The adversary $A$ adaptively queries the following oracles:

The **user register** oracle $O_{Reg}(\cdot, \cdot)$ takes the user identity $uid$ along with its public key $pk_{uid}$, then assigns a leaf node of $BT$ to store $(uid, pk_{uid})$ if $uid$ has not been queried in this oracle and returns $BT$ to $A$.

The **node key generation** oracle $O_{NK}(\cdot, \cdot)$ takes the identity identifier $uid$ and the attribute set $S_{uid}$ as input, and then runs the algorithm $NKGen(GP, uid, S_{uid}, BT)$ and returns the running result $NK_{uid}$ to $A$.

The **update key generation** oracle $O_{UK}(\cdot, \cdot)$ takes the time $t$ as input, then runs the algorithm $UKGen(GP, revl, rejf, BT, t)$ and returns $UK_t$ to $A$.

The **user revocation** oracle $O_{Rev}(\cdot, \cdot)$ takes the time $t$ and the identity $uid$ as input, then runs $UserRev(revl, uid, t)$ and returns the updated revocation list $rejf$ to $A$.

**Challenge** The adversary $A$ submits a challenge access matrix $(M^*, \rho^*)$, the time $t^*$ and two symmetric keys $KEY_0$ and $KEY_1$ with the same length to the challenger $C$. Among them, the challenge access matrix and $(M^*, \rho^*)$ time $t^*$ need to satisfy the following constraints:

1. For any query $O_{NK}(uid, S_{uid}^*)$ such that the attribute set $S_{uid}^*$ satisfies the challenge matrix, Then $uid$ must have been queried in $O_{Rev}(uid, t)$, where $t \leq t^*$.
2. For any non-revoked user $uid$, if its attribute set $S_{uid}^*$ satisfies the challenge matrix, then $uid$ must be in $O_{NK}(\cdot, \cdot)$ and has not been queried.

Then $C$ tosses a coin to determine the value of $\gamma \in \{0, 1\}$, then executes $Encrypt(KEY_{\gamma}, (M^*, \rho^*), GP, t^*) \rightarrow CT^*$ and returns the challenge ciphertext $CT^*$ to $A$.

**Query phase 2** After receiving the challenge ciphertext $CT^*$, $A$ continues to query the oracles according to Query Phase 1 adaptively.

**Guess** $A$ outputs a guess bit $\gamma' \in \{0, 1\}$ for $\gamma$. The winning advantage for the adversary is $A$ defined as $Adv_A = Pr[\gamma = \gamma'] - \frac{1}{2}$.

**Definition 4** A Revocable Attribute-Based Encryption EHR Sharing Scheme With Multiple Authorities(MA-RABE) in
blockchain is selective IND-CPA secure, if the advantages of all \((k, n)\) probabilistic polynomial time adversaries in the above game are negligible.

5 Construction

5.1 System initialization

AAs select the security parameter \(\lambda\) and the maximum number of users \(N\) as input, and execute the algorithm \(\text{Setup}(\lambda, N)\). The algorithm generates two cyclic groups \(\mathbb{G}\) and \(\mathbb{G}_t\) with a prime \(p\) order, where \(g\) is the generator of \(\mathbb{G}\). It also generates bilinear mapping \(\hat{\epsilon} : \hat{\epsilon}(\mathbb{G}, \mathbb{G}) \to \mathbb{G}_t\), an anti-collision function \(H : \{0, 1\}^* \to \mathbb{G}\) and two random group elements \(h\) and \(d \in \mathbb{G}_t\).

Suppose \(n\) is the total number of AAs, and \(k\) is the threshold for PSSS. \(n\) AAs generate shares \(\{a_i, b_i\}_{i \in \mathbb{N}}\) about random numbers \(a, b\) through PSSS. Then each authority calculates locally and submits \(\hat{\epsilon}(g, g^a, g^b)\) to the contract. The contract receives \(\hat{\epsilon}(g, g)^a, g^b\) sent by \(k\) authorities, and calculates as follows:

\[
\hat{\epsilon}(g, g)^a = \prod_{i=1}^{k}(\hat{\epsilon}(g, g)^a)^{\alpha_i} = \hat{\epsilon}(g, g)^{\sum_i \alpha_i d_i}
\]

\[
g^b = \prod_{i=1}^{k}(g^b)^{\beta_i} = g^{\sum_i \beta_i d_i}
\]

Therefore, the global parameter \(GP = \{\hat{\epsilon}, \mathbb{G}, \mathbb{G}_t, h, d, \hat{\epsilon}(g, g)^a, g^b, H\}\). Then contract generates a binary tree \(BT\) for DU, assigns a random number \(\xi_\theta \in G\) to each node \(\theta\) of \(BT\) and initializes two empty lists \(\text{revl}\) and \(\text{rejl}\) that can store \(N\) tuples.

5.2 User management

- **Register** Each DU generates its own private key \(usk \in \mathbb{Z}_p^*,\) and calculates \(g^usk\) as its public key. Then DU calls the registration function in the contract and sends his public key. After the registration function collects at least \(k\) AAs’ signatures, it will randomly select a \(BT\) leaf node to store \((uid, g^usk)\).

- **Revocation** AAs calls the revocation function in the contract with the parameter \((uid, t)\). After the revocation function collects at least \(k\) AAs’ signatures, it sets \((uid, \ast)\) to \((uid, t)\) in the list \(\text{rejl}\).

- **Rejoining** AAs calls the rejoining function in the contract with the parameter \((uid, t)\). After the rejoining function collects at least \(k\) AAs’ signatures, it sets \((uid, \ast)\) to \((uid, t)\) in the list \(\text{rejl}\).

**Remark** The input time to perform revocation and rejoining is always later than the time of the latest \(UK_t\). Otherwise, revocation and rejoicing of the user are meaningless. Similarly, when executing the revocation of \(uid\), the input time must be later than the time of \(uid\) in \(\text{rejl}\). When executing the rejoining of \(uid\), the input time must be later than the time of \(uid\) in \(\text{rejl}\).

5.3 Key generation

**Attribute recovery key assignment** Attribute recovery key is a part of the one-way anonymous key agreement used to recover the attributes in the matrix. DU calculates \(H(x)^{usk}\) for each owned attribute \(x\). Note that the attribute \(x\) here contains both the attribute name and the attribute value. Then DU sends \(H(x)^{usk}\) to each authority through a secure channel. To verify the ownership correctness, each authority needs to check:

\[
\hat{\epsilon}(H(x), pk_{uid}) = \hat{\epsilon}(H(x)^{usk}, g)
\]

After checking that it is correct, each attribute authority calculates the share of attribute recovery key token \(H(x)^{usk, \beta_i}\) and sends it to the contract. The contract collects at least \(k\) shares of the attribute recovery key token to recover attribute recovery key token \(\text{ARKT}_{uid,x}\) by interpolation:

\[
\text{ARKT}_{uid,x} = \prod_{i=1}^{k} H(x)^{usk, \beta_i} \cdot d_i
\]

DU decrypts \(\text{ARKT}_{uid,x}\) with its private key to get the attribute recovery key:

\[
\text{ARK}_x = \text{ARKT}_{uid,x}^{1/usk} = (H(x)^{usk, \beta_i})^{1/usk} = H(x)^\beta
\]

**Node key generation** AAs generate the node key \(NK_{uid}\) for \(uid\) based on the user attribute set \(S_{uid}\) and runs \(\text{NKGen}(GP, uid, S_{uid}, BT)\). The detailed process is as follows:

1. AAs retrieve and parse the leaf node \(\theta_{uid}\) that stores \(uid\) in \(BT\) to get the tuple \((uid, g^usk)\). For each node \(\theta\) in \(\text{Path}(\theta_{uid})\), AAs extract the value \(\xi_\theta\).
n AAs leverage \((k, n)\) PSSS to generate shares \(\{r_{\theta,i},v_{\theta,i},\omega_{\theta,i}\}_{\omega_{\theta,i} \in \mathcal{S}_{\theta}}\) of random numbers \(r_{\theta},v_{\theta},\omega_{\theta} \in \mathbb{G}\) for the user \(uid\).

3. For each node \(\theta\) in \(\text{Path}(\theta_{uid})\): each authority calculates locally and submits \(p_{\theta_{uid}}^{a_{\theta}}, z_{\theta}, d_{\theta}, g_{\theta}^{a_{\theta}}\) and \(\{g_{\theta_{uid}}^{a_{\theta}}, H(x)^{a_{\theta}}\cdot h_{\theta}^{a_{\theta}}, \cdot h_{\theta}^{\alpha_{\theta}}\}_{\alpha_{\theta} \in \mathcal{S}_{\theta_{uid}}}\) to the contract.

4. The contract receives shares of \(N_{uid}\) from at least \(k\) authorities, then calculates as follows:

\[
N_{\theta,1} = \prod_{i=1}^{k} (p_{\theta_{uid}}^{a_{\theta}}\cdot z_{\theta}^{\alpha_{\theta}} \cdot d_{\theta}^{\alpha_{\theta}})^{L(i)}
= p_{\theta_{uid}}^{k} \cdot z_{\theta}^{\sum_{i=1}^{k} \alpha_{\theta}} \cdot d_{\theta}^{\sum_{i=1}^{k} \alpha_{\theta}}\ 
= p_{\theta_{uid}}^{a_{\theta}} \cdot z_{\theta}^{\alpha_{\theta}} \cdot d_{\theta}^{\alpha_{\theta}}
\]

\[
N_{\theta,2} = \prod_{i=1}^{k} g_{\theta_{uid}}^{a_{\theta}} \cdot L(i)
= g_{\theta_{uid}}^{\sum_{i=1}^{k} a_{\theta} \cdot L(i)}
= g^{a_{\theta}}
\]

For each attribute \(x \in \mathcal{S}_{uid}\), it calculates:

\[
N_{\theta,x,1} = \prod_{i=1}^{k} (H(x)^{a_{\theta,x_i}} \cdot h_{\theta,x_i})^{L(i)}
= H(x)^{\sum_{i=1}^{k} a_{\theta,x_i} \cdot L(i)} \cdot h_{\theta,x_i}^{\sum_{i=1}^{k} a_{\theta,x_i} \cdot L(i)}
= H(x)^{a_{\theta,x_i}} \cdot h_{\theta,x_i}
\]

Therefore, the node key of \(uid\) is \(N_{uid} = \{N_{\theta,1}, N_{\theta,2}, N_{\theta,x,1}, N_{\theta,x,2}\}_{\omega_{\theta} \in \mathcal{S}_{\theta}}\).

**Update key generation** AAs run \(UKGen(GP, revl, rejL, BT, t)\) according to the revocation list, rejoin list and the time mark \(t\) to generate the update key \(UK_{t}\) at \(t\). The process is as follows:

1. AAs calculate the minimum covered node set \(\text{KUNode} (\text{rejl}, \text{rejL}, \text{BT}, t)\) according to the revocation list and the rejoining list.

2. \(n\) AAs leverage \((k, n)\) PSSS to generate shares \(\{s_{\theta,i}\}_{\omega_{\theta,i} \in \mathcal{S}_{\theta}}\) of random numbers \(s_{\theta} \in \mathbb{G}\) for the user \(uid\).

3. For each node \(\theta\) in \(\text{KUNode}(\text{rejl}, \text{rejL}, \text{BT}, t)\): each authority calculates locally and submits \(H(t)^{s_{\theta}} \cdot z_{\theta}^{-s_{\theta}}\) and \(g^{s_{\theta}}\) to the contract.

4. The contract receives shares of \(UK_{t}\) from at least \(k\) nodes, then calculates as follows:

\[
U_{\theta,1} = \prod_{i=1}^{k} (H(t)^{s_{\theta}} \cdot z_{\theta}^{-s_{\theta}})^{L(i)}
= H(t)^{\sum_{i=1}^{k} s_{\theta} \cdot L(i)} \cdot z_{\theta}^{-\sum_{i=1}^{k} s_{\theta} \cdot L(i)}
= H(t)^{s_{\theta}} \cdot z_{\theta}^{-s_{\theta}}
\]

\[
U_{\theta,2} = \prod_{i=1}^{k} g^{s_{\theta}} \cdot L(i)
= g^{\sum_{i=1}^{k} s_{\theta} \cdot L(i)}
= g^{s_{\theta}}
\]

The update key for legal users is \(UK_{t} = \{U_{\theta,1}, U_{\theta,2}\}_{\omega_{\theta} \in \text{KUNode}(\text{rejl}, \text{rejL}, \text{BT}, t)}\). Note that the input \(t\) for this algorithm is always the current moment, which is always later than any time in \(\text{rejL}\) and \(\text{rejL}\).

**Pre-decryption key generation** This process is usually completed by the CSP. \(uid\) sends a pre-decryption request to the CSP. Then the CSP retrieves the leaf node \(\theta_{uid}\) corresponding to \(uid\) in \(\text{BT}\). If \(\text{Path}(\theta_{uid}) \cap \text{KUNode}(\text{rejl}, \text{rejL}, \text{BT}, t) = \emptyset\), then the CSP returns rejection. Otherwise, according to the property of the KUNode algorithm, the non-revoked user’s \(\text{Path}(\theta_{uid})\) must have an intersection node \(\theta\) with \(\text{KUNode}(\text{rejl}, \text{rejL}, \text{BT}, t)\). The pre-decryption key of \(uid\) is calculated as follow:

\[
PDK_{\theta,uid} = N_{\theta,1} \cdot U_{\theta,1}
= p_{\theta_{uid}}^{a_{\theta}} \cdot z_{\theta}^{\alpha_{\theta}} \cdot d_{\theta}^{\alpha_{\theta}} \cdot H(t)^{s_{\theta}} \cdot z_{\theta}^{-s_{\theta}}
\]

Remark After the ARK, and \(N_{uid}\) of the user \(uid\) are distributed, the relationships between attribute \(x \in \mathcal{S}_{uid}\) and related keys can be represented by Fig. 3, in which the solid line represents the public relationship disclosed in the contract, and the dotted line represents the relationship known only to \(uid\) and AAs. And due to computational difficulties, others cannot deduce the user’s attribute information through the key information.

**5.4 Encryption**

As shown in Fig. 4, MA-RABE adopts a combination of symmetric and asymmetric hybrid encryption. The DO encrypts the data to share with the AES symmetric key and then encrypts the symmetric key by LSSS. The workflow of Encryption is shown in Fig. 4. The detailed process is as follows:
**Symmetric encryption** DO generates AES key $KEY \in \mathbb{G}_t$ and encrypts data by $\text{AESEncrypt}(KEY, data) \rightarrow CT_{KEY}$.

**Asymmetric encryption** DO customizes an access policy $AP$, where $J$ is the number of attributes in $AP$. Then DO converts $AP$ into the share-generating matrix $M_{dom}$ and the mapping function $\rho$ over the related attributes.

Then DO obfuscates the mapping function to fully hide the related attributes and protect the privacy of the data-sharing parties by the following three steps:

1. DO randomly selects a number $r \in Z_p^*$ and calculates $\sigma_j = \hat{e}(g^\rho_j, H(\text{attr}_j))$ for $j \in [J]$, where $\text{attr}_j$ is an attribute in the access policy.

2. DO computes the matrix confusion key $g^\rho$, which is used to recover part of the mapping function $\rho$ for $DU$ later.

3. DO replaces the previous mapping function $\rho(i)$ with $\rho'(i) = \hat{e}(g^\rho_j, H(\text{attr}_j))$ for $i \in [l]$.

Thus the access policy $AP$ is eventually converted into the LSSS scheme $\Pi: (M_{dom}, \rho')$ on the related attributes.

Next, DO runs $\text{Encrypt}(KEY, (M, \rho'), GP, t)$ to encrypt the AES key $KEY$ as the following steps:

1. DO obtains the time $t$ of the latest $UK_t$ from the contract, then computes $\bar{C} = KEY \cdot \hat{e}(g, g)^{\rho'}$ and $C_0 = H(t)^{\rho}$.

2. DO randomly selects $s, r_2, r_3, \ldots, r_m \in Z_p^*$ to form a vector $\bar{n} = \{s, r_2, r_3, \ldots, r_m\}^T$.

3. For $i \in [l]$, DO calculates $\lambda_i = M_i \cdot \bar{n}$, where $M_i$ is the $i$th row of matrix $M$.

4. DO randomly selects $\{u_i\}_{i \in [l]} \in Z_p^*$, then compute the remaining ciphertext as follows for $i \in [l]$:

\[
\begin{align*}
C_{i,1} &= g^{\lambda_i}, \\
C_{i,2} &= g^{s_i}, \\
C_{i,3} &= d^{\lambda_i} \cdot h^{s_i}, \\
C_{i,4} &= H(x)^{-\lambda_i}.
\end{align*}
\]

Finally, DO uploads the outsourced ciphertext $CT = ((M, \rho'), g^\rho, C_{i,1}, C_{i,2}, C_{i,3}, C_{i,4})_{i \in [l]}$ to the cloud server $CSP$.

### 5.5 Pre-Decryption

To alleviate the computational pressure caused by bilinear maps, CSP can pre-decrypt for $DU$ without leaking privacy and return the pre-decrypted result $CT'$. The workflow of **Pre-Decryption** is shown in Fig. 5. CSP performs pre-decryption by executing the following steps:

1. DO obtains the $(M, \rho'), g^\rho$ of $CT$ from CSP, then computes $\{\sigma'_j : \sigma'_j = \hat{e}(g^\rho_j, H(\text{attr}_j))\}_{j \in [S]}$, where $S$ is the number of the attribute set $S_{\text{id}}$.

\[
\text{PreDec}(KEY, PK_{\text{dom}}, N_{\\text{id}}, UK_t) \
\quad \text{Send } CT' \text{ to } DU
\]

---

**Fig. 3** The relationships between attribute $x$ and related keys

**Fig. 4** The workflow of Encryption. There are three entities involved in Encryption, DO means Data Owner, SC means Smart Contract, and DU means Data User.

**Fig. 5** The workflow of Pre-Decryption. There are three entities involved in Pre-Decryption, DO means Data Owner, SC means Smart Contract, and DU means Data User.

© Springer
2. DU computes the set of rows \( I = \{i : \{\sigma'_i\}_{\forall i \in [5]} \cap \{\rho'(i)\}_{\forall i \in [1]} \} \). This means that the DU recovers part of the mapping function \( \rho \) and knows which rows in the matrix \( M \) correspond to the attributes it has. Then DU sends the set \( I \) and the corresponding \( \{H(\rho(i))^{usc} \cdot \}_{\forall i \in I} \) to CSP.

3. If the attribute set of \( uid \) satisfies the access policy, then \( \{M_i\}_{\forall i \in I} \) must be a full-rank matrix. Therefore, the CSP can find a set of constants \( c_i \in \mathbb{Z}_p^* \) in a polynomial time such that \( \sum_{i \in I} c_i M_i = (1, 0, ..., 0) \).

4. Then CSP runs \( \text{PreDec}(CT, PDK_{\text{aid}}, NK_{\text{aid}}, UK_i) \). It tries to compute the pre-decryption key \( PDK_{\text{aid}} \) against \( NK_{\text{aid}} \) and \( UK_i \) stored in blockchain. If the calculation of the pre-decryption key fails, the rejection is returned to DU. Otherwise, the CSP finds \( \{N_{\theta,\rho(1),1}, N_{\theta,\rho(2)2}\}_{\forall i \in I} \) that corresponds to \( \{H(\rho(i))^{usc} \cdot \}_{\forall i \in I} \) as shown in Fig. 3 and computes:

\[
\prod_{i \in I} \left[ \frac{\hat{e}(C_{i,1}, PDK_{\text{aid}}) \cdot \hat{e}(N_{\theta,\rho(1),1}, C_{i,2}) \cdot \hat{e}(N_{\theta,\rho(2)2}, C_{i,2})}{\hat{e}(N_{\theta,2}, C_{i,3})} \right]^{c_i} \cdot \frac{1}{\hat{e}(C_{i,0}, U_{g,2})} = \hat{e}(g, g)^{usc-a_s} = CT'
\]

and further returns \( CT' \) to DU.

### 5.6 User decryption

DU runs \( \text{UserDec}(CT', sk_{\text{aid}}) \) locally after receiving \( CT' \).

1. DU uses its own private key to perform one exponentiation on the received \( CT' \) to obtain the symmetric key \( KEY \):

\[
KEY = \frac{\hat{C}}{(CT')^{usc}} \quad (16)
\]

2. Then, DU uses \( KEY \) to perform AES decryption on \( CT_{\text{KEY}} \) by \( \text{AESDecrypt}(KEY, CT_{\text{KEY}}) \rightarrow \text{data} \).

### 5.7 Ciphertext update

When the set of non-revoked users in the system changes (that is, user revocation or rejoining occurs), AAs will generate a key update message \( UK_{\text{latest}} \) about the latest moment. DO needs to update the \( C_0 \) in the \( CT \) according to the time of \( UK_{\text{latest}} \), so that users who are non-revoked at \( t_{\text{latest}} \) can obtain the plaintext.

\[
C_0 \leftarrow H(t_{\text{latest}}) \quad (17)
\]

### 6 Security Analysis

#### 6.1 Correctness

The correctness proof of Eqs. (15) and (16) is presented in Appendix A.

#### 6.2 Security proof

**Theorem 1** The MA-RABE scheme is IND-CPA secure if the DBDH problem is hard to break and \((k, n)\) PSSS is IND-CPA secure.

**Proof** Assume that there is a PPT adversary \( \mathcal{A} \) who can break the MA-RABE scheme in selective security model defined in IV.D with a non-negligible advantage \( \epsilon \). Given an instance of the DBDH problem \((g, g^a, g^b, g^z)\) where \( \mathcal{A} \) needs to decide \( Z = g^{ab} \) or \( Z = g^c \). And \( z \) is randomly selected from \( \mathbb{Z}_p^* \). There are two types of adversary who will query oracles and return the decision to the challenger \( \mathcal{C} \):

- \( \mathcal{A} \) is a legal user. \( \mathcal{A} \) cannot query the node key about \( S^* \), but \( \mathcal{A} \) can obtain the update key of the challenge time \( r^t \).
- \( \mathcal{A} \) is an illegal user. \( \mathcal{A} \) can query the node key about \( S^* \) and obtain the update key of the challenge of \( r^t \). But all users whose attribute sets satisfy the challenge matrix \((M^*, \rho^*) \) are revoked at \( r^t \) or before \( r^t \).

**Init** The challenger \( \mathcal{C} \) runs the adversary \( \mathcal{A} \). \( \mathcal{A} \) chooses the challenge matrix \((M^*, \rho^*) \) and sends it to \( \mathcal{C} \).

**Setup** The adversary \( \mathcal{A} \) specifies a corrupted authority set \( \mathcal{A}' \) with the size of \( n' \leq k - 1 \) so that the challenger \( \mathcal{C} \) can generate the global parameters \( \mathcal{GP} = [\hat{e}, G, \bar{G}, h, \bar{h}, e(g, g)^a, g^b, H] \). The hash function \( H \) is abstracted into \( H(y) = g^r \), \( \tau_e \in \mathbb{Z}_p^* \). Then \( \mathcal{C} \) returns \( \mathcal{GP} \) to \( \mathcal{A} \), where \( a = ab \) is implicitly defined for \( \hat{e}(g, g)^a = \hat{e}(g^a, g^b) \).

**Query phase 1** \( \mathcal{A} \) queries the following oracles adaptively.

\( \mathcal{O}_{\text{Req}}(\text{uid}, pk_{\text{aid}}) \): The adversary \( \mathcal{A} \) submits the identity \( \text{uid} \) along with the corresponding \( pk_{\text{aid}} \). The challenger \( \mathcal{C} \) picks an unsigned leaf node \( \theta_{\text{aid}} \) from \( BST \) and stores \((\text{uid}, pk_{\text{aid}})\) in \( \theta_{\text{aid}} \).

\( \mathcal{O}_{\text{Neg}}(\text{uid}, S_{\text{aid}}^*) \): The adversary \( \mathcal{A} \) submits the identity \( \text{uid} \) and the attribute set \( S_{\text{aid}}^* \). The challenger \( \mathcal{C} \) answers the query \((\text{uid}, S_{\text{aid}}^*)\) as follows:
Case 1 If $uid$ is a non-revoked user, and the attribute set $S^*_{uid}$ satisfies the challenge matrix $(M^*, \rho^*)$, then $O_{NK}(uid, S^*_{uid})$ cannot be queried.

Case 2 If $uid$ is a revoked user, and the attribute set $S^*_{uid}$ does not satisfy the challenge matrix $(M^*, \rho^*)$, then the challenger $C$ runs $NKGen(GP, uid, S^*_{uid}, BT)$ and returns the $NK^*_{uid}$ to $A$.

$O_{UK}(t)$: The adversary $A$ submits the identity $uid$ and a time mark $t$ to query the update key. Then the challenger $C$ runs $UKGen(GP, uid, rejl, BT, t)$ and returns the $UK^*_{uid}$ to $A$.

$O_{Rev}(uid, t)$: The adversary $A$ submits a time mark $t$ and the identity $uid$ to revoke $uid$. Then challenger $C$ runs revocation function with $(uid, t)$ and returns the updated $rejl$.

Note that the input time in this oracle must be later than the time of $uid$ in $rejl$.

**Challenge** $A$ submits two symmetric keys $KEY_0$ and $KEY_1$ with the same length and the challenge time $t^*$. Then $C$ tosses a coin to determine the value of $\gamma \in \{0, 1\}$ and computes $C = KEY_\gamma \cdot \hat{e}(g, g)^{\gamma abs} \cdot C_0 = (g^\gamma)^{\gamma abs}$ and $C' = g^\gamma$. For each row $M_i^*$ of $M^*$, $C$ computes

$$
\begin{align*}
C_{i, 1} &= g^d^i \\
C_{i, 2} &= g^{h^i} \\
C_{i, 3} &= d^{h^i} \cdot h^{h^i} \\
C_{i, 4} &= (g^\gamma)^{h^i}
\end{align*}
$$

Therefore, $CT^* = ((M^*, \rho^*), g^\gamma, CT_{KEY}, C, C_0, C', \{C_{i, 1}, C_{i, 2}, C_{i, 3}, C_{i, 4}\}_{\gamma \in \{0, 1\}})$. Finally, $C$ returns the challenge ciphertext $CT^*$ to $A$.

**Query phase 2** After receiving the challenge ciphertext $CT^*$, $A$ continues to query the oracles according to Query Phase 1 adaptively.

**Guess** $A$ outputs $\gamma' \in \{0, 1\}$ to guess the value of $\gamma$. $C$ returns 1 if $\gamma = \gamma'$ which denotes $Z = \hat{e}(g, g)^{\gamma abs}$. Otherwise, $C$ returns 0 which denotes $Z$ is a random number $g^\gamma$ in group $G$.

When $Z = \hat{e}(g, g)^{\gamma abs}$, the challenge ciphertext $CT^*$ is hard to break, $Pr[\gamma = \gamma'] = \frac{1}{2} + \epsilon$. When $Z = g^\gamma$, $Pr[\gamma = \gamma'] = \frac{1}{2}$. Therefore, the probability of $C$ to solve DBDH problem $Pr_{\gamma, DBDH} = \frac{1}{2}Pr[\gamma = \gamma']Z = \hat{e}(g, g)^{\gamma abs} + \frac{1}{2}Pr[\gamma = \gamma']Z = g^\gamma \frac{1}{2} + \epsilon - \frac{1}{2}$.

Since the probability for any PPT adversary to break DBDH problem is negligible. The probability of $C$ to solve DBDH problem $\frac{1}{2}$ is also negligible. We can derive that the advantage of $A$ to win the game $\epsilon$ is negligible. So the scheme MA-RABE achieves chosen-plaintext security.

### 6.3 Robustness

Since the scheme MA-RABE uses PSSS $(k, n)$ to distribute user keys and manage users, so its robustness is affected by the threshold $k$ and the total number of AAs $n$. We assume that the attacker can make some AAs in MA-RABE go down. We suppose the probability that a single AA goes down is $\delta$. When the attack further expands, until the number of AAs work is less than $k$, it will make the system crashed directly. So the probability of system crash satisfies the Bernoulli distribution $\sum^n_{i=k+1} \binom{n}{i} \delta^i (1 - \delta)^{n-i}$. Figure 6a, b show the probability of system crash versus the threshold of PSSS $k$ and the probability of a single AA crash $d$ when $n$ is 5 and 10 respectively.

Next, we consider another situation. If the attacker controls at least $k$ AAs in MA-RABE through some means and thus controls the entire system. We assume that the probability of a single AA being controlled is $c$, then the probability of system being controlled also satisfies the Bernoulli distribution $\sum^n_k \binom{n}{i} c^i (1 - c)^{n-i}$. Figure 7a, b show the probability of the system being controlled versus the threshold of PSSS $k$ and the probability of a single AA being controlled $c$ when $n$ is 5 and 10 respectively.

From the analysis of Figs. 6 and 7, we can see that compared to a system based on a single AA, the MA-RABE has better robustness. The closer that $k$ gets to $n$ in the scheme,
the more chance the system crashes, and the probability of 
the system being compromised gets smaller. On the contrary, 
the less \( k \) is than \( n \), the less chance the system crashes, and 
the probability of the system being controlled will be higher. 
On the other hand, when \( k \) is larger, the computational over-
head of reconstructing the secret will increase. Therefore, 
choosing appropriate \( k \) and \( n \) can simultaneously ensure low 
computational overhead and strong security, such as (3,5), 
(5,10), (6,10).

6.4 Decentralization

The AAs in MA-RABE adopts PSSS protocol to negotiate 
to generate keys and manage users. This process requires 
at least \( k \) authorities. Any single authority cannot affect the 
security of the protocol, nor will it affect the security of the 
system. Therefore, there is no central entity in the applica-
tion layer of the system. In addition, the user’s status and 
key information are stored in the smart contract, and the 
correctness of the operation result of the smart contract is 
verified by all full nodes of the blockchain. Therefore, there 
is no central entity in the code layer of the system. In sum-
mary, MA-RABE achieves decentralization and effectively 
resists single points of failure.

6.5 Collusion resistance

According to the decryption process in Appendix A, if the 
colluders want to decrypt the ciphertext with their attributes, 
they need to specify the private key of the mastercolluder. 
Therefore, it is necessary to use a PDK containing the mas-
tercolluder’s private key. The \( v_0 \) in \( N_{\theta,2} \) is the same as the 
v_0 in PDK, and it belongs to the same user under the same 
node. Therefore, this process also needs to fix the mastercol-
luder’s \( N_{\theta,2} \). In addition, \( C_{i,1}, C_{i,2}, C_{i,3}, C_{i,4}, C_0 \) and \( U_{\theta,2} \) are 
also need to be fixed.

Recall the generation process of \( N_{\theta,\rho(i),2} \). For the same 
node, the \( v_0 \) for different users are different. Therefore, 
when the colluders share their \( N_{\theta,\rho(i),2} \), the \( \hat{e}(h^v, g^u) \) in 
the numerator cannot be eliminated by the \( \hat{e}(g^v, h^u) \) in the 
denominator.

6.6 Policy privacy

In the process of encrypting the AES key, DO uses the one-
way anonymous key to replace the explicit mapping function 
\( \rho(i) \) between the rows of the matrix and attributes with \( \rho(i)' \): 
\( \rho'(i) = \hat{e}((g^\theta)^j, H(\text{attr}_i)) \). In order to recover the rows of the 
matrix to attributes, the decryptor has two methods:

1. The decryptor needs to know the value of \( r \). However, 
the discrete logarithm problem in cyclic groups is dif-
ficult to solve. The decryptor cannot derive the value of 
\( r \) from the public \( g^r \).
2. According to the bilinearity of Bilinear Maps: 
\( \hat{e}(g^{r'}, H(\text{attr}_j)^\theta) = \hat{e}((g^\theta)^j, H(\text{attr}_i)) \). This process requires 
\( g^{r'} \) and \( H(\text{attr}_j)^\theta \). The matrix confusion key \( g^{r'} \) is part of 
the ciphertext and is public. \( H(\text{attr}_j)^\theta \) is the ARK_{attr} 
of DU. So only the decryptor with the \{ARK_{attr,i}\}_{i\in[l]} \) can 
recover the rows of the matrix to attributes.

In addition, recall that the CSP will receive the pre-decryption 
request information the set of rows \( l \) and the correspond-
ing \{H(\rho'(i))_{\rho'\in[l]}\}_{i\in[l]} \) from DU. Among them, \( H(\rho'(i))_{\rho'\in[l]} \) is the 
ARK_{\rho'(i)} of DU, which is public in the contract and 
is associated with \{N_{\theta,\rho'(i),1}, N_{\theta,\rho'(i),2}\}_{i\in[l]} \). CSP cannot derive 
attribute information.

Therefore, our scheme MA-RABE realizes policy privacy 
protection.

7 Performance evaluation

This section compares the proposed scheme MA-RABE with 
other CP-ABE schemes in terms of function implementation, 
storage and computational overhead.

![Fig. 7 The probability of system being controlled varies with the probability of one single authority being controlled](image-url)
7.1 Security and functionality comparison

We make a comparison between previous CP-ABE schemes and our scheme in Table 1 in terms of the attribute authority working mechanism, the underlying blockchain architecture, user rejoining, user revocation, hidden policy, and outsourcing decryption. The scheme of [44] has an assumed credible CA and thus has the problem of centralized power. In the scheme of [20], each AA manages an attribute set which is non-intersecting to others, so the breakdown of any AA will affect the availability of attribute management. In other schemes, the attribute Universe is jointly managed by a group of attribute authorities, which can prevent single points of failure and enhance system security. In addition, the proposed scheme MA-RABE has richer user management functions and supports hidden policy and outsourcing pre-decryption, which can greatly reduce the computing overhead on the user side without leaking user privacy.

7.2 The storage overhead

In this section, we compare the proposed MA-RABE with other schemes in terms of the storage overhead. The used notions in comparison are listed in Table 2. \( N \) and \( n \) respectively denote the number of users and AA in the scheme.

From the Table 3, we can see that the central authority appears in the schemes [44] and [17], where the CA key of [44] is a constant size and the CA key of [44] is positively correlated with the number of AA, the number of users and the number of attributes. The other three schemes [20, 32] and MA-RABE do not involve the central authority, so there is no need to consider the CA storage burden caused by the scheme’s scale and complexity.

In addition, we can see that the scheme [32] has the lowest storage overhead on the DU side. This is because the scheme [32] does not process the LSSS matrix for privacy protection, and the DU only needs to store the user private key used to decrypt the symmetric key with the size of \( |Z_p^*| \). Since the proposed MA-RABE adopts the one-way anonymous key agreement, which requires additional storage of the corresponding attribute recovery key for each attribute. Therefore, the storage overhead on DU side in this solution is \( A_u |G| + |Z_p^*| \).

In terms of ciphertext storage overhead, since all schemes use AES symmetric encryption and ABE asymmetric encryption, the symmetrically encrypted ciphertexts in these schemes all require \( |\text{SymCT}| \)-sized space. In addition, the storage overhead of the ciphertext is related to the number of parameters set. It is reasonable that the ciphertext storage overhead of MA-RABE is slightly more than other schemes under the condition of ensuring privacy and realizing ciphertext pre-decryption. Finally, in terms of updating ciphertext, only the scheme [20] and MA-RABE realize the ciphertext update function. When MA-RABE updates the ciphertext, the ciphertext size that CSP needs to update only needs \(|G|\), which is much smaller than scheme [20].

### Table 1 The Comparison of Functionality

| Schemes    | Authority working mechanism       | Blockchain | User rejoining | User revocation | Hidden policy | Outsourcing Pre-decryption |
|------------|------------------------------------|------------|----------------|-----------------|---------------|----------------------------|
| [44]       | Single, Centralization             |            |                |                 |               |                            |
| [17]       | Multiple, Jointly collaborating     |            |                |                 |               |                            |
| [20]       | Multiple, Exclusive Sets            | ✓          | ✓              | ✓               | ✓             |                            |
| [32]       | Multiple, Jointly collaborating     | ✓          | ✓              | ✓               | ✓             | ✓                          |
| MA-RABE    | Multiple, Jointly collaborating     | ✓          | ✓              | ✓               | ✓             | ✓                          |

“Single, Centralization” means that there is only one Central Authority in the scheme. “Multiple, Exclusive Sets” means that there are multiple Attribute Authorities that each manage a disjoint attribute set in the scheme. “Multiple, Jointly collaborating” means that there are multiple Attribute Authorities that jointly manage the Attribute Universe in the scheme. ✓ indicates the existence of the functionality in the scheme.
7.3 The computational overhead

In this section, we compare the proposed MA-RABE with other schemes in terms of the computational overhead. The used notions in comparison are listed in Table 2. We assume that $R$ is the number of revoked users and $k$ is the secret sharing threshold in the multi-AA scheme.

Table 4 compares the computational overhead in detail, focusing on the most time-consuming operations: bilinear pairing operation and exponentiation operation.

From Table 4 we can know that when encrypting data on the DO side, the scheme [20] has fewer exponentiation operations in $G$ and more exponentiation operations in $G_t$ than scheme [32]. While the MA-RABE performs more exponentiation operations in $G$ are used than scheme [32].

In terms of DU-side decryption, since the scheme [32] and MA-RABE both adopt outsourcing pre-decryption, the decryption overhead of these two schemes is lower than the scheme [20].

In terms of the cost of updating the key, the scheme [20] needs to update the decryption key for each non-revoked user. Besides, the scheme [32] needs to update the attribute key for each non-revoked user, so it has the most computational overhead. In contrast, MA-RABE has the lowest key update overhead, which is unrelated to the number of user attributes.

In terms of the cost of updating ciphertext, the scheme [20] and MA-RABE implement the ciphertext update function, and we can clearly see that MA-RABE has a lower and constant computational cost for updating ciphertext.

To further evaluate the actual performance of the schemes, we performed the simulation on a Ubuntu system with AMD R5 4600H CPU 3.0 GHz and 16 GB RAM. The cyclic group in the experiment adopts the curve group of the PBC library based on Python. The smart contracts are deployed on Hyperledger Fabric. The experimental data are an average of 50 simulation results.

Figure 8a contains the simulation results of encrypting the symmetric key on the DO side. The horizontal axis is the number of attributes in the access policy, i.e., the number of rows of the access matrix; the vertical axis is the time spent on encryption. Since the scheme [20] has more exponentiation operations, encryption takes significantly more time than the other two schemes. However, when encrypting the symmetric key, the scheme [32] performs fewer exponentiation operations in $G$ by the number of attributes than MA-RABE, so it takes slightly less time than MA-RABE.

Figure 8b presents the decryption overhead of the symmetric key on the DU side. The horizontal axis is the number of attributes in the access policy; the vertical axis is the decryption time. Assuming that before decryption, DU has calculated the $\hat{e}(g', H(x)^{\beta})$ of attributes required for decryption against the matrix confusion key $g'$. For the same $g'$, the $\hat{e}(g', H(x)^{\beta})$ only needs to be calculated once forever. As shown in Figure 8b, since the pre-decryption in MA-RABE is outsourced to the CSP, DU only needs to perform one exponentiation operation to obtain the symmetric key, which has extremely high decryption efficiency.

| Table 3 | The comparison of storage performance |
|---------|--------------------------------------|
| Waters [44] | $2Z_t^1$ | $(2A_{E}+4)Z_t^1$ | $\text{SymCT}+(2A_{E}+1)G+|G_t|$ | - |
| Li et al. [17] | $(2T_{E}+2N+n+10)Z_t^p$ | $(2A_{E}+5)Z_t^p$ | $\text{SymCT}+(2A_{E}+1)G+|G_t|$ | - |
| Zhong et al. [20] | - | $2A_{E}G+|Z_t^p|$ | $\text{SymCT}+(3A_{E}+1)G+|G_t|$ | $A_{E}|G_t|$ |
| Liu et al. [32] | - | $|Z_t^p|$ | $\text{SymCT}+(3A_{E}+2)G+|G_t|$ | - |
| MA-RABE | - | $A_{E}G+|Z_t^p|$ | $\text{SymCT}+(4A_{E}+2)G+|G_t|$ | $|G|$ |

| Table 4 | The comparison of computational performance |
|---------|----------------------------------|
| Encryption on DO side | Decryption on DU side | Update of Key | Update of Ciphertext |
| Zhong et al. [20] | $(3A_{E}+1)E+2A_{E}E_{E}+E_{E}$ | $A_{E}(E_{E}+2P)+E_{E}$ | $(N-R+1)E+E_{E}$ (off-chain) | $A_{E}P$ |
| Liu et al. [32] | $(4A_{E}+2)E+E_{E}+E_{E}$ | $E_{E}+E_{E}$ | $k(N-R)(2A_{E}+2)E$ (on-chain) | - |
| MA-RABE | $(5A_{E}+2)E+E_{E}+E_{E}$ | $E_{E}+E_{E}$ | $k(RlongN)R$ (on-chain) | $E$ |
Figure 8c shows the calculation results of the ciphertext update on the DO side. The horizontal axis is the number of attributes in the access policy; the vertical axis is the time spent in the calculation. The overhead of the scheme [20] has a linear relationship with the number of attributes. Since MA-RABE only needs to perform an exponentiation operation on updating the ciphertext, it has a constant ciphertext update overhead, which is obviously better than the former.

In general, the scheme has slightly more encryption calculation overhead on the DO side than the scheme [32]. But compared with the schemes [20] and [32], MA-RABE has sufficient advantages in other aspects.

8 Conclusion

In this article, we propose a secure revocable ABE scheme with multiple authorities MA-RABE based on blockchain. This solution can well solve the problem of privacy leakage and computational burden in the EHR sharing process. First, multiple authorities leverage PSSS ($k,n$) to manage users and distribute keys to prevent centralized power and achieve strong security robustness. Besides, MA-RABE supports any LSSS policy hiding through the novel decentralized one-way anonymous key agreement, which effectively protects the privacy of data owners and data users. The MA-RABE also achieves key update through the BT with low overhead. Considering that the EHR sharing scenario involves devices with limited computing resources, outsourcing decryption is introduced into MA-RABE. Theoretical analysis and experimental simulation show that the MA-RABE can implement more complex functions and meet the requirements of low storage overhead and low user decryption overhead. The future work for MA-RABE is to implement verifiable outsourcing decryption to prevent the CSP from dishonestly executing the protocol and returning to DU the wrong intermediate decryption ciphertext.

Appendix A. The correctness proof of equations

The CSP pre-decrypts the ciphertext for DU as Eq. (15). The correctness proof of the Eq. (15) is presented here.
The DU decrypts the pre-decryption result locally as Eq. (16). The correctness of Eq. (16) is presented as follows.

\[
CT' = \prod_{i \in I} \left[ \frac{\hat{e}(C_{i,1}, PDK_{\theta, \text{aud}}) \cdot \hat{e}(N_{\theta, \phi(i), 1}, C_{i,1}) \cdot \hat{e}(N_{\theta, \phi(i), 2}, C_{i,2})}{\hat{e}(N_{\theta, 2}, C_{i,3})} \right]^{\gamma_i} \cdot \frac{1}{\hat{e}(C_{i,0}, C_{i,2})}
\]

\[
= \prod_{i \in I} \left[ \frac{\hat{e}(g^h, pk_{\text{aud}} \cdot d_{x^i} \cdot H(t)^{x_i}) \cdot \hat{e}(g^{\alpha(i) \cdot \omega(i)}, H(x)^{-\omega(i)}) \cdot \hat{e}(H(x)^{\alpha(i) \cdot \omega(i)}, h^{\omega(i)}) \cdot \hat{e}(h^{\alpha(i) \cdot \omega(i)}, g^{\omega(i)})}{\hat{e}(g^{\omega(i)}, d_{x^i}) \cdot \hat{e}(g^{\omega(i)}, h^{\omega(i)}) \cdot \hat{e}(H(t)^{\omega(i)}, g^{\omega(i)})} \right]^{\gamma_i} \cdot \frac{1}{\hat{e}(H(t)^{\omega(i)}, g^{\omega(i)})}
\]

\[
= \prod_{i \in I} \left[ \hat{e}(g^{\alpha(i) \cdot \omega(i)}, H(t)^{x_i}) \cdot \hat{e}(g^{\omega(i)}, d_{x^i}) \cdot \hat{e}(g^{\omega(i)}, h^{\omega(i)}) \cdot \hat{e}(H(t)^{\omega(i)}, g^{\omega(i)}) \right]^{\gamma_i} \cdot \frac{1}{\hat{e}(H(t)^{\omega(i)}, g^{\omega(i)})}
\]

\[
= \prod_{i \in I} \left[ \hat{e}(g^{\alpha(i) \cdot \omega(i)}, H(t)^{x_i}) \cdot \hat{e}(g^{\omega(i)}, d_{x^i}) \cdot \hat{e}(g^{\omega(i)}, h^{\omega(i)}) \cdot \hat{e}(H(t)^{\omega(i)}, g^{\omega(i)}) \right]^{\gamma_i} \cdot \frac{1}{\hat{e}(H(t)^{\omega(i)}, g^{\omega(i)})}
\]

\[
= \frac{\hat{e}(g^{\alpha(i) \cdot \omega(i)}, H(t)^{x_i}) \cdot \hat{e}(g^{\omega(i)}, d_{x^i}) \cdot \hat{e}(g^{\omega(i)}, h^{\omega(i)}) \cdot \hat{e}(H(t)^{\omega(i)}, g^{\omega(i)})}{\hat{e}(H(t)^{\omega(i)}, g^{\omega(i)})}
\]

\[
= \hat{e}(g, g)^{\omega(i) - x_i}
\]

\[
KEY = \frac{C}{(CT')^{1/x_{\text{ask}}}}
\]

\[
= \frac{KEY \cdot \hat{e}(g, g)^{\omega(i) - x_i}}{\hat{e}(g, g)^{\omega(i) - x_i}}
\]

\[
= \frac{KEY \cdot \hat{e}(g, g)^{\omega(i) - x_i}}{\hat{e}(g, g)^{\omega(i) - x_i}}
\]

\[
= \hat{e}(g, g)^{\omega(i) - x_i}
\]

Funding This work is supported by the National Key R&D Program of China under Grant 2017YFB0802300 and the Natural Science Foundation of Hebei Province under Grant F2021201052.

Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Shi S, He D, Li L, Kumar N, Khan MK, Choo KKR (2020) Applications of blockchain in ensuring the security and privacy of electronic health record systems: a survey. Comput Secur 97:101966. https://doi.org/10.1016/j.cose.2020.101966

2. Madhavan S, Bastarache L, Brown JS, Butte AJ, Dorr DA, Embl PJ, Friedman CP, Johnson KB, Moore JH, Kohane IS et al (2021) Use of electronic health records to support a public health response to the Covid-19 pandemic in the United States: a perspective from 15 academic medical centers. J Am Med Inform Assoc 28(2):393–401. https://doi.org/10.1093/jamia/ocaa287

3. Kheer S, Moniruzzaman M, Yassine A, Benlamri R (2019) Blockchain technology in healthcare: a comprehensive review and directions for future research. Appl Sci 9(9):1736. https://doi.org/10.3390/app9091736

4. Donawa A, Orukari I, Baker CE (2019) Scaling blockchains to support electronic health records for hospital systems. In: 2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON), IEEE, pp 0550–0556. https://doi.org/10.1109/UEMCON47517.2019.8993101

5. Chukwu E, Garg L (2020) A systematic review of blockchain in healthcare: Frameworks, prototypes, and implementations. IEEE Access 8:21196–21214. https://doi.org/10.1109/ACCESS.2020.2969881

6. Zhang J, Yang Y, Liu X, Ma J (2022) An efficient blockchain-based hierarchical data sharing for healthcare internet of things. IEEE Transactions on Industrial Informatics, pp 1–1. https://doi.org/10.1109/TII.2022.3145851

7. Hao J, Huang C, Ni J, Rong H, Xian M, Shen XS (2019) Fine-grained data access control with attribute-hiding policy for cloud-based IoT. Comput Netw 153:1–10. https://doi.org/10.1016/j.comnet.2019.02.008

8. Xiang X, Zhao X (2022) Blockchain-assisted searchable attribute-based encryption for E-health systems. J Syst Archit 124:102417. https://doi.org/10.1016/j.sysarc.2022.102417

9. Sun P (2020) Security and privacy protection in cloud computing: Discussions and challenges. J Netw Comput Appl 160. https://doi.org/10.1016/j.jnca.2020.102642

10. Zhang G, Chen X, Feng B, Guo X, Hao X, Ren H, Dong C, Zhang Y (2022) BCST-APTS: Blockchain and CP-ABE empowered data supervision, sharing, and privacy protection scheme for secure and trusted agricultural product traceability system. Secur Commun Netw 2022:2958963. https://doi.org/10.1155/2022/2958963

11. Agili SF, Sedaghat M, Singelée D, Gupta M (2022) MLS-ABAC: Efficient multi-level security attribute-based access control scheme. Futur Gener Comput Syst 131:75–90. https://doi.org/10.1016/j.future.2022.01.003

12. Chase M (2007) Multi-authority attribute based encryption. In: Theory of cryptography conference, Springer, pp 515–534. https://doi.org/10.1007/978-3-540-70936-7_28

13. Chase M, Chow SS (2009) Improving privacy and security in multi-authority attribute-based encryption. In: Proceedings of the 16th ACM conference on Computer and communications security, pp 121–130. https://doi.org/10.1145/1653662.1653678
14. Li Q, Zhang Q, Huang H, Zhang W, Chen W, Wang H (2022) Secure, efficient and weighted access control for cloud-assisted industrial IoT. IEEE Internet Things J 1–1. https://doi.org/10.1109/JIOT.2022.3146197

15. Zhang Y, Deng RH, Xu S, Sun J, Li Q, Zheng D (2020) Attribute-based encryption for cloud computing access control: a survey. ACM Comput Surv (CSUR) 53(4):1–41. https://doi.org/10.1145/3398036

16. Li J, Chen X, Chow SS, Huang Q, Wong DS, Liu Z (2018) Multi-authority fine-grained access control with accountability and its application in cloud. J Netw Comput Appl 112:89–96. https://doi.org/10.1016/j.jnca.2018.03.006

17. Li W, Xue K, Xue Y, Hong J (2015) TMACS: a robust and verifiable threshold multi-authority access control system in public cloud storage. IEEE Trans Parallel Distrib Syst 27(5):1484–1496. https://doi.org/10.1109/TPDS.2015.2448095

18. Liu Q, Liu Y, Luo M, He D, Wang H, Choo KKR (2022) The security of blockchain-based medical systems: Research challenges and opportunities. IEEE Syst J 1–12. https://doi.org/10.1109/jsyst.2022.3155156

19. Mayer AH, da Costa CA, Righi RDR (2020) Electronic health records in a blockchain: a systematic review. Health Inform J 26(2):1273–1288. https://doi.org/10.1177/1460458219866350

20. Zhong H, Zha W, Xu Y, Cui J (2018) Multi-authority attribute-based encryption access control scheme with policy hidden for cloud storage. Soft Comput 22(1):243–251. https://doi.org/10.1007/s00500-016-2330-8

21. Sarra R, Kumar C, Barbhuiya FA (2022) Macf: a multi-authority access control scheme with efficient ciphertext and secret key size for fog-enhanced IoT. J Syst Archit 123:102347. https://doi.org/10.1016/j.sysarc.2021.102347

22. Qin X, Huang Y, Yang Z, Li X (2021) A blockchain-based access control scheme with multiple attribute authorities for secure cloud data sharing. J Syst Archit 112. https://doi.org/10.1016/j.sysarc.2020.101854

23. Ramu G (2018) A secure cloud framework to share EHRs using modified CP-ABE and the attribute bloom filter. Educ Inf Technol 23(5):2213–2233. https://doi.org/10.1007/s10639-018-9713-7

24. Wu A, Zhang Y, Zheng X, Guo R, Zhao Q, Zheng D (2019) Efficient and privacy-preserving traceable attribute-based encryption in blockchain. Ann Telecommun 74(7–8):401–411. https://doi.org/10.1007/s12243-018-06499-y

25. Fan K, Xu H, Gao L, Li H, Yang Y (2019) Efficient and privacy preserving access control scheme for fog-enabled iot. Futur Gener Comput Syst 99:134–142. https://doi.org/10.1016/j.future.2019.04.003

26. Beimel A et al (1996) Secure schemes for secret sharing and key distribution. https://www.cs.bgu.ac.il/~beimel/Papers/thesis.pdf

27. Zhang Z, Zhang J, Yuan Y, Li Z (2021) An expressive fully policy-hidden ciphertext policy attribute-based encryption scheme with credible verification based on blockchain. IEEE Internet Things J 1–1. https://doi.org/10.1109/JIOT.2021.3177378

28. Yang L, Li C, Cheng Y, Yu S, Ma J (2022) Achieving privacy-preserving sensitive attributes for large universe based on private set intersection. Inform Sci 582:529–546. https://doi.org/10.1016/j.ins.2021.09.034

29. Wei J, Chen X, Huang X, Hu X, Susilo W (2021) RS-HABE: Revocable-storage and hierarchical attribute-based access scheme for secure sharing of e-health records in public cloud. IEEE Trans Dependable Secure Comput 18(5):2301–2315. https://doi.org/10.1109/TDSC.2019.2947920

30. Xiong H, Zhao Y, Peng L, Zhang H, Yeh KH (2019) Partially policy-hidden attribute-based broadcast encryption with secure delegation in edge computing. Futur Gener Comput Syst 97:453–461. https://doi.org/10.1016/j.future.2019.03.008

31. Zheng D, Qin B, Li Y, Tian A (2020) Cloud-assisted attribute-based data sharing with efficient user revocation in the internet of things. IEEE Wirel Commun 27(3):18–23. https://doi.org/10.1109/MWC.001.1900433

32. Liu S, Yu J, Xiao Y, Wan Z, Wang S, Yan B (2020) BC-SABE: Blockchain-aided searchable attribute-based encryption for cloud-IoT. IEEE Internet Things J 7(9):7851–7867. https://doi.org/10.1109/JIOT.2020.2993231

33. Guo R, Yang G, Shi H, Zhang Y, Zheng D (2021) O 3-R-CP-ABE: an efficient and revocable attribute-based encryption scheme in the cloud-assisted IoMT system. IEEE Internet Things J 8(11):8949–8963. https://doi.org/10.1109/JIOT.2021.3055541

34. Zhang R, Li J, Lu Y, Han J, Zhang Y (2022) Key escrow-free attribute based encryption with user revocation. Inform Sci 600:59–72. https://doi.org/10.1016/j.ins.2022.03.081

35. Zhang J, Li T, Jiang Q (2022) Ma J (2022b) Enabling efficient traceable and revocable time-based data sharing in smart city. EURASIP J Wirel Commun Netw 1:3. https://doi.org/10.1186/s13638-021-02072-5

36. Pedersen TP (1991) A threshold cryptosystem without a trusted party. In: Workshop on the Theory and Application of of Cryptographic Techniques, Springer, pp 522–526. https://doi.org/10.1007/3-540-46416-6_47

37. Shamir A (1979) How to share a secret. Commun ACM 22(11):612–613. https://doi.org/10.1145/359168.359176

38. Deepa N, Pham QV, Nguyen DC, Bhattacharya S, Prabadevi B, Gadekallu TR, Maddikunta PKR, Fang F, Pathirana PN (2022) A survey on blockchain for big data: Approaches, opportunities, and future directions. Futur Gener Comput Syst 131:209–226. https://doi.org/10.1016/j.future.2022.01.017

39. Lin SY, Zhang L, Li J, Li Ji, Sun Y (2022) A survey of application research based on blockchain smart contract. Wireless Netw 28(2):635–690. https://doi.org/10.1007/s11276-021-02874-x

40. Zou J, He D, Zeadally S, Kumar N, Wang H, Choo KR (2021) Integrated blockchain and cloud computing systems: a systematic survey, solutions, and challenges. 54(8). https://doi.org/10.1145/3456628

41. Yang X, Li W (2020) A zero-knowledge-proof-based digital identity management scheme in blockchain. Comput Secur 99. https://doi.org/10.1016/j.cose.2020.102050

42. Han D, Pan N, Li KC (2022) A traceable and revocable ciphertext-policy attribute-based encryption scheme based on privacy protection. IEEE Trans Dependable Secure Comput 19(1):316–327. https://doi.org/10.1109/TDSC.2020.2977646

43. Rasori M, La Manna M, Perazzo P, Dini G (2022) A survey on attribute-based encryption schemes suitable for the internet of things. IEEE Internet Things J 1–1. https://doi.org/10.1109/JIOT.2022.3154039

44. Waters B (2011) Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization. In: International Workshop on Public Key Cryptography, Springer, pp 53–70. https://doi.org/10.1007/978-3-642-19379-8_4

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.
Xiaohui Yang is a Professor at the School of Cyber Security and Computer, Hebei University, China. He received his PhD degree from the University of Science and Technology of China in 2010. He has published several research papers in reputed international journals and conferences. His primary research interests include distributed computing, information security and trusted computing.

Wenjie Li received his M.S. degree from the School of Cyber Security and Computer, Hebei University of China in 2021. Currently, he is a Ph.D. student at the School of Cyber Engineering, Xidian University, China. He is working on Cryptography and Blockchain technology novel applications, such as blockchain-based systems, decentralized applications, smart contracts programming and security and software development. His major research interests include information security, peer-to-peer networks, cryptocurrencies and blockchains.

Kai Fan received the B.S., M.S., and Ph.D. degrees in telecommunication engineering, cryptography and telecommunication and information system from Xidian University, Xi’an, China, in 2002, 2005, and 2007, respectively. He is currently an Associate Professor with the State Key Laboratory of Integrated Service Networks, Xidian University. He has authored/co-authored over 70 papers in journals and conferences. He has received nine Chinese patents. He has managed five national research projects. His research interests include IoT security and information security.