FDRShare: A Fully Decentralized and Redactable EHRs Sharing Scheme with Constant-Size Ciphertexts *

Zhichao Li, Zhexi Lu, Lingshuai Wang, Qiang Wang, and Che Bian

***Abstract*—Blockchain-based Electronic Health Records (EHR) sharing schemes can enable the owner to outsource the encrypted EHR to the powerful but untrusted cloud such that only authorized users can search or access EHRs and check whether the cloud returns valid results or not. However, these existing schemes suffer from three substantial shortcomings that limit their usefulness: (i) a centralized Trusted Authority (TA) is introduced to manage keys, which fully conflicts with the decentralized nature of blockchain; (ii) the size of the encrypted EHRs is linear to the attribute set size, which poses challenges for the blockchain network; (iii) the efficiency of search seriously impacts the user experience, which hinders the deployment in practice. To address these issues, we proposed FDRshare, a fully decentralized EHRs sharing scheme with constant-size ciphertexts.**

***Index Terms*—Searchable Encryption, Blockchain, Attribute-Based Encryption**

I. Introduction

With the explosive increase of global information, cloud computing has been experiencing unprecedented development. To capture the market as much as possible, the cloud service providers have been rushing to launch their products, such as Amazon EC2 and S3, Microsoft Azure, and Google App Engine. Due to lower cost, higher reliability, better performance, and faster deployment, enterprises and individuals have been increasingly outsourcing their storage tasks to the cloud. As a typical and concrete application instance of cloud storage, the cloud-based electronic health record (EHR) has been playing a significant role in the healthcare industry. Unlike traditional paper-based health records, EHR can be shared among different institutions by outsourcing them to the cloud. It can vigorously facilitate personalized treatment plans, disease analysis, prediction, etc. However, past real-world incidents [1] and recent research [2] have shown that the cloud cannot be fully trusted and may expose sensitive data and forge the query result. To this end, several privacy-preserving EHR sharing schemes supporting integrity checks (PPShare) [3], [4] have been proposed. In terms of privacy, the owners encode their EHR with some specified access control policies using attribute-based encryption (ABE) before uploading to the cloud. The users such as doctors or researchers with different attributes send the search tokens to make the cloud search the target EHR over all encrypted EHRs. The user can decode them if and only if the attribute set satisfies the policies embedded into ciphertexts. In terms of integrity, authenticated data structure (ADS) such as accumulator [5] or Merkel hash tree [6] is utilized to check the correctness of the query results with an overwhelming probability.

However, most of the existing PPShare schemes [7], [8] rely on the centralized cloud to manage EHR and respond to the users' query requests. In such a centralized model, it inevitably suffers from DDoS attacks and single-point failure. To overcome this challenge, Hu et al. [9] proposed a novel blockchain-based scheme, which utilizes smart contracts to manage and search EHRs rather than the cloud. Given a search token, each consensus node has to faithfully execute search operations through the smart contract. If not, everyone can find his misbehavior. Due to its importance, several blockchain-based EHR sharing schemes (BBShare) [10]–[14] have been proposed over the past decade. However, these schemes still cannot be applied in practice. The main reasons are summarized as follows:

Conflict between Centralized TA and Blockchain: Most of the existing BBShare schemes [10], [11], [13], [15] make use of ciphertext policy ABE (CP-ABE) [16] to facilitate fine-grained sharing of EHRs. However, CP-ABE depends on a centralized trusted authority (TA) for key management and attribute distribution. This seriously contradicts the decentralized nature of blockchain. In addition, the security of the schemes will be compromised if the fully trusted TA is corrupted. Hence, it is essential to achieve full decentralization.

Unbearable Storage Overhead: To protect EHR privacy, the existing BBShare schemes employ the CP-ABE to encode their EHR indexes and store the encrypted indexes on the blockchain. However, a common issue is that most of them rely on the traditional CP-ABE scheme, where the size of ciphertext increases at least linearly with the number of attributes involved in the access policy. This creates a conflict with the inherent storage limitations of blockchain, severely restricting its practicality. To address this issue, some schemes [12], [14], such as MedShare and BBF, have been proposed to replace the traditional CP-ABE with CP-ABE with constant-size ciphertext. However, it is worth noting that these schemes still depend on a centralized TA as mentioned above. Hence,

Z. Chao, Z. Lu and Q. Wang are with Software College, Northeastern University, China.

L. Wang is with School of Computer Science and Engineering, Northeastern University, China.

C. Bian is with The Fourth Affiliated Hospital, China Medical University, China.

Corresponding author: Qiang Wang. E-mail: wangqiang1@mail.neu.edu.cn.

it is essential to design a fully decentralized CP-ABE with constant-size ciphertexts.

Search Efficiency Challenges: Most of the existing BBshare schemes [10]–[12] are interactive, where the user needs to interact with the cloud during the search phase. Multiple-round interactions impose a serious impact on the efficiency of search. To tackle this issue, Wang et al. [14] introduced the first non-interactive BBshare scheme called Medshare based on the work [17]. To the best of our knowledge, Medshare is the only non-interactive scheme achieving sublinear search performance, in line with the standard of plaintext information retrieval algorithms. However, it requires users to conduct $O(n \cdot q)$ exponentiation operations, where n is the number of matches of the least frequent keyword in the boolean search and q is the number of terms in the search. These exponentiation operations not only exhibit low efficiency but also cost high gas fees on the blockchain. Hence, it is essential to design a lightweight search method that eliminates computationally intensive operations such as exponentiation.

A. Our Results and Contributions

To resolve the problems mentioned above, we propose, FDRShare, a fully decentralized EHRs sharing scheme with constant-size ciphertexts. Our results and contributions can be summarized as follows:

- To achieve decentralized fine-grained access control with constant-size ciphertexts, we propose a decentralized constant-size CP-ABE (DCS-ABE) scheme, which employs multiple attribute authorities instead of one single TA for key management and utilizes an AND^*_m access control structure to generate the constant-size ciphertext.
- To improve search efficiency, we integrated a bloom filter into the non-interactive blockchain boolean search protocol. The Bloom filter is utilized to initially filter out the irrelevant search results, thereby preventing them from being processed by the search algorithm executed by the smart contract.

II. Related Work

Electronic health record (EHR) makes a transition from paper to digital medical records. This transition has accelerated health information sharing since it makes the use of them more flexible. To alleviate the maintenance burden of massive EHRs, the existing EHR sharing schemes commonly resort to the powerful cloud. Nevertheless, as the cloud is untrusted, it inevitably suffers from two major security challenges: privacy and integrity. To overcome these challenges, Nayak at el. initialed the first privacy-preserving EHR sharing scheme supporting integrity checks [18]. Following this pioneering work, plenty of schemes have been proposed. According to their architectures, it can be roughly categorized into two folds: centralized ones [19] and decentralized ones [10], [14], [20]. The former relies on the centralized cloud to manage EHR and handle query requests. In contrast, the latter is an enhanced version that incorporates security measures. It leverages blockchain techniques to resist DDoS attacks and single-point failure, which are potential vulnerabilities in the former. All search and upload operations over EHRs are performed through smart contracts rather than the cloud. Due to the limitation of space, we only focus on the decentralized privacy-preserving EHR sharing schemes supporting integrity checks (DPPShare) in the following.

Azaria et al. [20] utilize searchable symmetric encryption (SSE) to construct the first DPPShare scheme. Since SSE allows only the private key holder (i.e., the owner) to produce ciphertexts and to create trapdoors for search, the owners cannot share their EHRs with others. To overcome this challenge, [11], [12] incorporated traditional CP-ABE with SSE schemes to achieve fine-grained access control. The users with different attributes send the search tokens to make the smart contract search the target EHR over all encrypted EHRs. However, traditional CP-ABE schemes, which introduce a TA for attribute authentication and key management, will make the system suffer from the single points of failure and compromise attacks. Additionally, the linear growth of CP-ABE ciphertext size with the attribute set's size imposes a significant storage burden on the blockchain.

To address these issues, we design a decentralized ABE scheme with a constant-size ciphertext, which supporting high-efficency on-chain search. Our theoretical comparison with the most advanced model proposed by Wang [14] is presented in TABLE I. Although our scheme shares the same time complexity with [14] in Search algorithm, our Search algorithm improves the search efficiency by avoiding time-consuming exponential or bilinear pairing operations.

TABLE I
Comparison of Computation Cost

Complexities	MedShare [14]	FDRShare
ParameterSetup	$O(1)$	$O(1)$
IndexGen	$O(n)$	$O(n)$
KeyGen	$O(k)$	$O(k)$
STGen	$O(q^2)$	$O(q)$
Search	$O(n \cdot q)$	$O(n \cdot q)$

III. PRELIMINARIES

A. Bilinear Pairing

Let $\mathbb{Z}_p$ be a finite field over prime p. Let $\mathbb{G}$, $\mathbb{G}_T$ be two cyclic multiplicative groups of the same prime order p. g is the generator of $\mathbb{G}$. Denote $\hat{e}$ a bilinear pairing map : $\mathbb{G}\times\mathbb{G} \rightarrow \mathbb{G}_T$ with the following properties:

1) Bilinear: $\forall a, b \in \mathbb{Z}_p$, $\hat{e}\left(g^a, g^b\right) = \hat{e}\left(g, g\right)^{ab}$.
2) Non-degenerate: $\hat{e}\left(g, g\right) \neq 1$.
3) Computable: There exists an efficient algorithm to compute $\hat{e}\left(g_1, g_2\right)$ for all $g_1, g_2 \in \mathbb{G}$.

B. DDH Assumption

Let $\mathbb{G}$ be a cyclic group of prime order p, the decisional Diffie-Hellman(DDH) problem is to distinguish the ensembles $\left\{(g, g^a, g^b, g^{ab})\right\}$ from $\left\{(g, g^a, g^b, g^z)\right\}$, where the elements $g \in \mathbb{G}$ and $a, b, z \in \digamma_p$ are chosen uniformly at random. We

say that the DDH assumption holds if for all efficient adversary $\mathcal{A}$, advantage

$$Adv_{\mathcal{A}}^{DDH}(1^{\lambda}) = \big|\Pr[\mathcal{A}(\mathbb{G}, p, g, g^a, g^b, g^c) = 1] - \Pr[\mathcal{A}(\mathbb{G}, p, g, g^a, g^b, g^{ab}) = 1]\big|$$

is negligible.

C. Pseudo-Random Function

Let F be a function transforming the element $x \in X$ to an output $y \in Y$ with a secret seed $k \in K_{prf}$. We say F is a pseudo-random function (PRF), if for all efficient adversaries $\mathcal{A}$, advantage

$$Adv_{F,\mathcal{A}}^{prf}(1^{\lambda}) = \Pr[A^{F(k,\cdot)}(1^{\lambda}) = 1] - \Pr[A^{f(\cdot)}(1^{\lambda}) = 1]$$

is negligible, where f is a truly random function from X to Y.

IV. SYSTEM DEFINITION

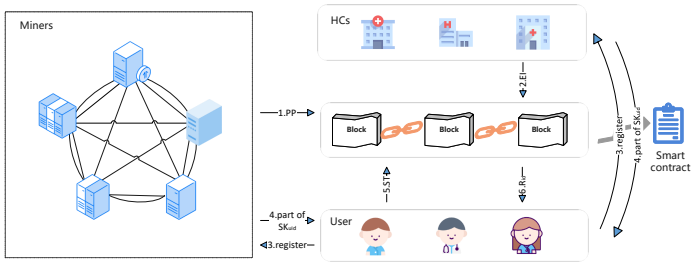

Fig. 1. The Architecture of FDRShare

The architecture of FDRShare is depicted in Figure 1. To begin with, Miners initiate the public parameters denoted as PP and disseminate them across the blockchain network. Subsequently, HCs employ the same PP to generate encrypted indexes, denoted as EI , which are then deployed onto the smart contract.

After the system setup, users enter the system and initiate registration requests directed toward both Miners and HCs. Following the authentication of these requests, Miners and HCs provide the respective users with sets of secret keys. Armed with these secret key sets, users are able to generate encrypted search tokens and effectively engage with the smart contract to retrieve search results. Users can employ their secret key sets to decrypt the obtained search results. Successful decryption is contingent upon whether users' attributes fulfill the access policies integrated within the encrypted search results.

A. Threat Model

We make the following threat assumptions for three entities in our system:

- HCs(Healthcare Centers) are always honest. As the builders and sharers of EHR, they try their best to protect the confidentiality of EHR. They are responsible for generating and distributing authorized keyword search keys for users. Besides, they generate the encrypted indexes and deploy them to the smart contract according to the protocol.
- Miners are honest. We use the term "Miners" to refer to the consensus nodes in the blockchain. They are responsible for generating public parameters of the whole system. Meanwhile, they execute the tasks of authorizing attributes and issuing secret keys for users, as well as faithfully executing search contract based on a consensus algorithm.
- Users are potential adversaries. They are doctors, nurses, and so on, who are allowed to access the EHR. While they generate search tokens honestly according to the protocol, they intend to extract sensitive information from encrypted indexes, query transactions, and search results. Moreover, they attempt to collude with each other to gain access privileges beyond their identity.

B. System Model

Formally, the system model of FDRShare consists of the following four algorithms:

1) $(\mathsf{PP}, \mathsf{MK}) \leftarrow \mathsf{Setup}\left(1^{\lambda}, \mathsf{DB}\right)$: The system setup algorithm is run by HCs and Miners. Given the security parameter $1^{\lambda}$and local EHR database DB as inputs, this algorithm outputs the public parameters PP and the system master key MK.
2) $\mathsf{SK}_{\mathsf{uid}} \leftarrow \mathsf{KeyGen}\left(\mathsf{PP}, \mathsf{MK}, \mathsf{uid}\right)$: The key generation algorithm is run by Miners and HCs jointly. Given the public parameters PP, the system master key MK, and the user identity uid as inputs, this algorithm outputs the users' secret key set $\mathsf{SK}_{\mathsf{uid}}$.
3) $\mathsf{ST} \leftarrow \mathsf{STGen}\left(\mathsf{SK}_{\mathsf{uid}}, \mathsf{Q}, \mathsf{PP}\right)$: The search token generation algorithm is run by Users and smart contract jointly. Given the users' secret key set $\mathsf{SK}_{\mathsf{uid}}$, the query set Q, and the public parameters PP as inputs, this algorithm outputs the search token ST.
4) $\mathsf{R}_{\mathsf{id}} \leftarrow \mathsf{Search}\left(\mathsf{PP}, \mathsf{ST}\right)$: The smart contract runs the on-chain search algorithm. Given the public parameters PP and the search token ST as inputs, this algorithm outputs the result set R_{id}.

V. BUILDING BLOCK

Before delving into the detailed construction of FDRShare, we first introduce our detailed construction of $\mathsf{DSC\text{-}ABE}$ in this section. To resolve the tackles posed by TA and the unbearable storage overhead brought by traditional CP-ABE, we design a DCS-ABE scheme, which employs multiple authorities and has constant-size ciphertext, based on the MA-ABE scheme and constant-size CP-ABE scheme [21], [22]. In EHR sharing application scenario, we consider that users have multiple attributes, and each of them has multiple values. We employ a set of pre-selected consensus nodes (miners) that serve as attribute authorities. Each Miner manages a set of mutually exclusive attributes. Let $\mathsf{A}_{\mathsf{k}} = \{\mathsf{i}_1, \mathsf{i}_2, ..., \mathsf{i}_{\mathsf{n}_{\mathsf{ki}}}\}$

$(1 \leqslant k \leqslant n_{Miner})$ denote the set of attributes managed by the k-th Miner, and $i_k\,(1 \leqslant i_k \leqslant n_{ki})$ represent the i-th attribute managed by the k-th Miner. The set of attribute values managed by the k-th Miner is denoted as V_k , where $v_{k,n}$ represents the set of values for the n-th attribute managed by the k-th Miner. We specify j_{i_k} as the j-th value of the i-th attribute managed by the k-th Miner. The decentralizing constant-size CP-ABE $(DCS-ABE)$ scheme consists of the following four algorithms:

1) $(mpk, msk) \leftarrow DSCABE.Setup\left(1^{\lambda}\right)$: The DSC-ABE setup algorithm is run by Miners. Miners negotiate a secure parameter 1^{λ}as input to generate a collision-resistant hash function $H_0 : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$, system attribute set A_k , attribute value set V_k , and an integer multiplication cyclic group $\mathbb{G}$. The cyclic group $\mathbb{G}$ has an order of p and generator g . Additionally, each miner independently chooses a pair of random numbers $\alpha_k, \beta_k \epsilon \mathbb{Z}_N$ as their private key and stores them locally, and computes

$$M_{i_k,j_{i_k}} = g^{-H\left(\alpha_k||i_k||j_{i_k}\right)}, N_{i_k,j_{i_k}} = e\left(g,g\right)^{H_0\left(\beta_k||i_k||j_{i_k}\right)}. \tag{1}$$

The ABE public key is $mpk = \left\langle g, \left\{ M_{i_k,j_{i_k}}, N_{i_k,j_{i_k}} \right\} \right\rangle$, and the ABE master key is $msk = \{\alpha_k, \beta_k\}, 1 \leqslant k \leqslant n_{Miner}$. Given access policy $\mathbb{A}$, we can aggregate mpk according to access policy $\mathbb{A}$:

$$\langle M_{\mathbb{A}} , N_{\mathbb{A}} \rangle = \left\langle \prod_{i_k \in \mathbb{A}} \bar{M}_{i_k} , \prod_{i_k \in \mathbb{A}} \bar{N}_{i_k} \right\rangle, \tag{2}$$

where $\bar{M}_{i_k} = M_{i_k,j_{i_k}}$ and $\bar{N}_{i_k} = N_{i_k,j_{i_k}}$. We can utilize this structure for ABE encryption then.

2) $ASK_{uid} \leftarrow DCSABE.KeyGen\,(msk, uid)$: The DSC-ABE key generation algorithm is run by Miners. Given the master key msk and the user identity uid as inputs, this algorithm generates the attribute secret key ASK_{uid} . When users join the system, they securely transmit uid to all Miners for identity verification. Once the verification is successful, each miner provides the corresponding attribute secret key. The attribute secret key from the k-th miner, with attribute i_k and attribute value j_{i_k}, is represented as:

$$ask_{i_k,j_{i_k}} = g^{H_0\left(\beta_k||i_k||j_{i_k}\right)} H_2\left(uid\right)^{H_0\left(\alpha_k||i_k||j_{i_k}\right)}. \tag{3}$$

The users' secret attribute secret key is $ASK_{uid} = \left\{ ask_{i_k,j_{i_k}} \right\}$.

3) $\mathbb{C} \leftarrow DCSABE.Enc\,(mpk, m, \mathbb{A})$: The DSC-ABE encryption algorithm is run by HCs. Given the master key mpk , message m, and access control policy $\mathbb{A}$ as inputs, this algorithm outputs the ABE ciphertext $\mathbb{C}$. In FDRShare, the file is encrypted by the AES algorithm and then uploaded to IPFS. The AES secret key is denoted as K_f . For each file upload, IPFS returns a unique file identifier id . HCs input mpk and $\mathbb{A}$ to generate an access control structure $\langle M_{\mathbb{A}} , N_{\mathbb{A}} \rangle$, which is used to encrypt $m = id||K_f$. Finally, HCs select a random number $s \in \mathbb{Z}_P^*$ and encrypt the data:

$$\mathbb{C} = F_{id} = \langle C_0, C_1, C_2 \rangle, \tag{4}$$

where $C_0 = (id\,\|K_f\,) \cdot N_{\mathbb{A}}^s$, $C_1 = g^s$, $C_2 = M_{\mathbb{A}}^s$. Since attributes are aggregated into exponents, the ciphertext is constant-size.

4) $m \leftarrow DCSABE.Dec\,(uid, ASK_{uid}, \mathbb{C})$: The DCS-ABE decryption algorithm is run by Users. Given the user identity uid , attribute secret key ASK_{uid} , and ABE ciphertext $\mathbb{C}$ as inputs, this algorithm outputs the decrypted result $m = id||K_f$ or an $\perp$. The user aggregates their attribute secret key $ASK_{uid} = \prod_{i_k \in L_{uid}} ask_{i_k,j_{i_k}}$, where L_{uid} denotes the user's attribute set, and decrypts:

$$m = id\,\|K_f = \frac{C_0}{\hat{e}\left(ASK_{uid}, C_1\right) \cdot \hat{e}\left(H_2\left(uid\right), C_2\right)}. \tag{5}$$

If the user's attribute set L_{uid} satisfies the $\mathbb{A}$ embedded in F_{id} , or in other words, if $L \models \mathbb{A}$, the user can successfully decrypt and obtain the value $id\,\|K_f$. Otherwise, an $\perp$ will be returned. With result $id\,\|K_f$, the user can use the file identifier id to locate the corresponding file on IPFS and decrypt the file using the symmetric key K_f .

VI. DETAIL CONSTRUCTION OF FDRSHARE

The detailed construction of FDRShare is as follows:

1) $(PP, MK) \leftarrow Setup\left(1^{\lambda}, DB\right)$: The system setup algorithm consists of two sub-algorithms: ParameterSetup and IndexGen.

- $(PK, MK) \leftarrow ParameterSetup\left(1^{\lambda}\right)$: The parameter setup algorithm is run by Miners and HCs. Given 1^{λ}as input, Miners invoke DCSABE.Setup to generate the ABE public key mpk and the ABE master key msk. Then, Miners choose two cyclic groups $\mathbb{G}$ and $\mathbb{G}_{\mathbb{T}}$ with the same order p . g is the generator of $\mathbb{G}$. Based on the chosen groups, Miners selects the bilinear maps $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_{\mathbb{T}}$, $e_1 : \mathbb{G} \times \mathbb{G}_{\mathbb{T}} \rightarrow \mathbb{G}$, and three hash functions $H_0 : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$, $H_1 : \{0,1\}^* \rightarrow \{0,1\}^{\lambda}$, and $H_2 : \{0,1\}^* \rightarrow \mathbb{G}$. After that, Miners choose the pseudo-random function F and the invertible pseudo-random permutation function P .
  Receiving the public parameters, HCs randomly select a seed k and a generator $g_1 \in \mathbb{Z}_n^*$ for the pseudo-random function F . Then, HCs initiate the authorized keyword search keys $\mathbf{w} = \{w_1, w_2, ..., w_n\}$, where we map each keyword to a prime number w_k for efficient storage [23]. Part of the $\mathbf{w}$ will then be sent to authenticated Users for generating search tokens.
  The system public key is $PK = (H_0, H_1, H_2, F, P, msk)$, and the system master key is denoted as $MK = (p, g_1, k, msk, \mathbf{w})$. The order p and the ABE secret key msk are kept by Miners, while $\mathbf{w}$, g_1, and k are kept by HCs.

Algorithm 1:

Input: $IV\,(w)$, $FW\,(id)$, PK and the system $masker\ key\ MK$

Output: $EIindex\,\{AIindex, BFindex, PTindex\}$

Initialize set $AIindex$, $BFindex$, $PTindex$ to empty dict;
for *each* $w \in DB$ **do**
  Initialize counter int c = 0;
  $w_{t_0} \leftarrow \{0,1\}^{\lambda}$;
  for *all* $id \in IV(w)$ **do**
    Set the counter $c = c + 1$;
    Generate a random nonce $t_c \leftarrow \{0,1\}^{\lambda}$;
    Set $w_{t_c} \leftarrow P\left(t_c, w_{t_{c-1}}\right)$;
    Calculate $stagw\,(Eq.6)$;
    Set $l \leftarrow H_1\,(stagw)$;
    Calculate $F_{id}\,(Eq.4)$;
    Calculate $\Gamma_{id} = F_{id}||t_c \oplus H_2\,(l||w_{t_c})$;
    Set $\sigma \leftarrow H_1\,(stagw\,\|w_{t_c})$;
    Append $AIindex\,[\sigma] = \Gamma_{id}$;
  end
  Append $PTindex\,[l]\ =\ w_{t_c}\,\|c$;
end
for *each* id *in* DB **do**
  Initialize $\vec{V}_{BF}^{id}$ to empty vector;
  Initialize I_{id} to empty list;
  for *all* $w \in FW(id)$ **do**
    $\vec{V}_{BF}^{id} \leftarrow storage\ w\ into\ \vec{V}_{BF}^{id}$;
    calculate $I_w = g^{H_1(stagw)}$;
    Add I_w to I_{id};
  end
  Append $BFilter\,[F_{id}] = \vec{V}_{BF}^{id}||I_{id} = \mathcal{B}_{id}$;
end
Deploy $AIindex$, $BFindex$, $PTindex$ **to smart contract**

Algorithm 2: Search token generation

Input: User private key SK_{uid}, Search query set Q, Partial Token index $PTindex$

Output: Search Token ST

Initialized Trapdoor T to an empty array.;
Initialized $\vec{V}_{BF}^{Q}$ to a vector.;
Calculate $stagw_l$; Set $l \leftarrow H_1\,(stagw_l)$;
Get $w_{t_c}\,\|c \leftarrow PTindex\,[l]$;
for *each* $w/\{w_l\}$ *in* Q **do**
  $\vec{V}_{BF}^{Q} \leftarrow$ Store w to $\vec{V}_{BF}^{Q}$;
  Calculate $I_w = g^{H_1(stagw)}$;
  Add I_w to T;
end
return $ST\left\{\vec{V}_{BF}^{Q}, Trapdoor\ T, w_{t_c}\,\|c, stagw_l\right\}$

Algorithm 3:

Input: Search Token $ST\left\{\vec{V}_{BF}^{Q}, Trapdoor\ T, w_{t_c}\,\|c, stagw_l\right\}$, $AIindex$, $BFindex$

Output: Result set R_{id}

for $i = c$ *to 1* **do**
  Set $\sigma = H_1\,(stagw_l\,\|w_{t_c})$;
  Get $\Gamma_{id} \leftarrow AIindex.find\,(\sigma)$;
  $F_{id}\,\|t_c \leftarrow \Gamma_{id} \oplus H_2\,(stagw\,\|w_{t_c})$;
  $Get\left\{V_{BF}^{id}, I_{id}\right\} \leftarrow$BFindex.find($F_{id}$);
  if $V_{BF}^{Q}\ \cup\ V_{BF}^{id}\ !=\ V_{BF}^{id}$ **then**
    continue;
  else
    if $T\ \subseteq\ I_{id}$ **then**
      Add F_{id} to result R_{id};
    end
  end
  Set $w_{t_{i-1}} \leftarrow P^{-1}\,(t_i, w_{t_i})$;
end
return result R_{id};

- $\mathsf{EI} \leftarrow \mathsf{IndexGen}\,(\mathsf{DB}, \mathsf{PK}, \mathsf{MK})$: The index generation algorithm is run by HCs. First, HCs process DB to obtain $\mathsf{FW}\,(\mathsf{id})$ and $\mathsf{IV}\,(w)$. Specifically, $\mathsf{IV}\,(w)$ establishes a mapping from the keyword w to all identifiers id of files containing w , while $\mathsf{FW}\,(\mathsf{id})$ establishes a mapping from the file identifier id to the keywords contained in the corresponding file. HCs generate the encrypted indexes according to Algorithm 1. The HCs then update all the new indexes on the blockchain.

The final output of the system setup algorithm is the system public key $\mathsf{PP} = \{\mathsf{PK}, \mathsf{EI}\}$ and the system master MK.

2) $\mathsf{SK_{uid}} \leftarrow \mathsf{KeyGen}\,(\mathsf{PP}, \mathsf{MK}, \mathsf{uid})$: The system pubic key PP can be parsed as $\{\mathsf{PK}, \mathsf{EI}\}$. Given the msk from MK and the user identity uid , which are sent to Miners and HCs by Users when registering, as inputs, Miners invoke DCSABE.KeyGen to generate the users' secret attribute secret key $\mathsf{ASK_{uid}}$. Subsequently, HCs assign authorized user keyword search keys $\mathbf{w}_{uid}$, where $\mathbf{w}_{uid} \in \mathbf{w}$, and a keyword private key set $\mathsf{WSK} = \mathsf{g_1}^{\frac{1}{\prod_{i=1}^{n} w_i}}$ to the user based on uid . The users' secret key set is $\mathsf{SK_{uid}} = (\mathsf{k}, \mathbf{w_{uid}}, \mathsf{WSK}, \mathsf{ASK_{uid}})$. Finally, Miners and HCs transmit the secret key set to the user through a secure channel.

3) $\mathsf{ST} \leftarrow \mathsf{STGen}\,(\mathsf{SK_{uid}}, \mathsf{Q}, \mathsf{PP})$: PP can be parsed as $\{\mathsf{PK}, \mathsf{EI}\}$. Given the users' secret key set $\mathsf{SK_{uid}}$, boolean query Q , and the encrypted indexes EI as inputs, the user selects the least frequent word $\mathsf{w_l}$ from Q and calculates:

$$stagw = \mathsf{F}\left(\mathsf{k}, (\mathsf{WSK})^{\prod_{w \in \mathbf{w_i}/\{\mathsf{w_l}\}} w}\right), \tag{6}$$

and submits it to the smart contract. Subsequently, the smart contract executes Algorithm 2 and returns the search token ST .

4) $\mathsf{R_{id}} \leftarrow \mathsf{Search}(\mathsf{PP}, \mathsf{ST})$: The PP can be parsed as $\{\mathsf{PK}, \mathsf{EI}\}$. Given the encrypted indexes EI and the search token ST as inputs, the smart contract executes Algorithm 3.

VII. Conclusion

In this paper, we present FDRShare, a new blockchain-based EHR sharing system with fully decentralized fine-grained access control. Compared to the existing solutions, our solution offers a more efficient privacy-protecting search service. Additionally, we have designed the DCS-ABE scheme based on blockchain characteristics, achieving further decentralization. To the best of our knowledge, our solution is currently the only one that addresses both of these aspects, making it highly practical for real-world applications. Future research directions include further refining the proposed DCS-ABE scheme and on-chain searchable encryption scheme to enhance their security. Additionally, introducing editability into the system represents a valuable avenue of investigation, as it can facilitate the broader adoption of the system. In the initial full version of the paper, we designed a scheme for an editable blockchain system, which is not presented in this article due to space constraints.

Acknowledgment

We thank the anonymous reviewers for their fruitful suggestions. This work was supported in part by the National Natural Science Foundation of China under Grant 62202090 and 62173101, by Liaoning Province Natural Science Foundation Medical-Engineering Cross Joint Fund under Grant 2022-YGJC-24, by Doctoral Scientific Research Foundation of Liaoning Province under Grant 2022-BS-077, and by the Fundamental Research Funds for the Central Universities under Grant N2217009.

References

[1] R. Zhang and L. Liu, "Security models and requirements for healthcare application clouds," in *2010 IEEE 3rd International Conference on cloud Computing*. IEEE, 2010, pp. 268–275.

[2] M. Z. Hasan, M. Z. Hussain, Z. Mubarak, A. A. Siddiqui, A. M. Qureshi, and I. Ismail, "Data security and Integrity in Cloud Computing," in *2023 International Conference for Advancement in Technology (ICONAT)*, Jan. 2023, pp. 1–5.

[3] B. E. Reedy and G. Ramu, "A secure framework for ensuring ehr's integrity using fine-grained auditing and cp-abe," in *2016 IEEE 2nd International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing (HPSC), and IEEE International Conference on Intelligent Data and Security (IDS)*, 2016, pp. 85–89.

[4] Y. Su, J. Sun, J. Qin, and J. Hu, "Publicly verifiable shared dynamic electronic health record databases with functional commitment supporting privacy-preserving integrity auditing," *IEEE Transactions on Cloud Computing*, vol. 10, no. 3, pp. 2050–2065, 2020.

[5] W. I. Khedr, H. M. Khater, and E. R. Mohamed, "Cryptographic accumulator-based scheme for critical data integrity verification in cloud storage," *IEEE Access*, vol. 7, pp. 65 635–65 651, 2019.

[6] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, "Enabling public auditability and data dynamics for storage security in cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 5, pp. 847–859, 2011.

[7] M. Joshi, K. Joshi, and T. Finin, "Attribute based encryption for secure access to cloud based ehr systems," in *2018 IEEE 11th International Conference on Cloud Computing (CLOUD)*, 2018, pp. 932–935.

[8] M. Li, S. Yu, Y. Zheng, K. Ren, and W. 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.

[9] S. Hu, C. Cai, Q. Wang, C. Wang, Z. Wang, and D. Ye, "Augmenting Encrypted Search: A Decentralized Service Realization with Enforced Execution," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 6, pp. 2569–2581, Nov. 2021.

[10] Q. Xia, E. B. Sifah, K. O. Asamoah, J. Gao, X. Du, and M. Guizani, "MeDShare: Trust-Less Medical Data Sharing Among Cloud Service Providers via Blockchain," *IEEE Access*, vol. 5, pp. 14 757–14 767, 2017.

[11] P. Zhang, J. White, D. C. Schmidt, G. Lenz, and S. T. Rosenbloom, "Fhirchain: applying blockchain to securely and scalably share clinical data," *Computational and structural biotechnology journal*, vol. 16, pp. 267–278, 2018.

[12] S. Wang, Y. Zhang, and Y. Zhang, "A Blockchain-Based Framework for Data Sharing With Fine-Grained Access Control in Decentralized Storage Systems," *IEEE Access*, vol. 6, pp. 38 437–38 450, 2018.

[13] G. Wu, B. Zhu, and J. Li, "Bmks: A blockchain based multi-keyword search scheme for medical data sharing," in *2022 IEEE Symposium on Computers and Communications (ISCC)*. IEEE, 2022, pp. 1–7.

[14] M. Wang, Y. Guo, C. Zhang, C. Wang, H. Huang, and X. Jia, "MedShare: A Privacy-Preserving Medical Data Sharing System by Using Blockchain," *IEEE Transactions on Services Computing*, vol. 16, no. 1, pp. 438–451, Jan. 2023.

[15] G. Yu, X. Zha, X. Wang, W. Ni, K. Yu, P. Yu, J. A. Zhang, R. P. Liu, and Y. J. Guo, "Enabling Attribute Revocation for Fine-Grained Access Control in Blockchain-IoT Systems," *IEEE Transactions on Engineering Management*, vol. 67, no. 4, pp. 1213–1230, Nov. 2020.

[16] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *2007 IEEE symposium on security and privacy (SP'07)*. IEEE, 2007, pp. 321–334.

[17] D. Cash, S. Jarecki, C. Jutla, H. Krawczyk, M.-C. Roşu, and M. Steiner, "Highly-scalable searchable symmetric encryption with support for boolean queries," in *Advances in Cryptology–CRYPTO 2013: 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I*. Springer, 2013, pp. 353–373.

[18] S. K. Nayak and S. Tripathy, "Privacy preserving provable data possession for cloud based electronic health record system," in *2016 IEEE Trustcom/BigDataSE/ISPA*, 2016, pp. 860–867.

[19] S. Chenthara, K. Ahmed, H. Wang, and F. Whittaker, "Security and privacy-preserving challenges of e-health solutions in cloud computing," *IEEE Access*, vol. 7, pp. 74 361–74 382, 2019.

[20] A. Azaria, A. Ekblaw, T. Vieira, and A. Lippman, "Medrec: Using blockchain for medical data access and permission management," in *2016 2nd international conference on open and big data (OBD)*. IEEE, 2016, pp. 25–30.

[21] A. Lewko and B. Waters, "Decentralizing attribute-based encryption," in *Advances in Cryptology–EUROCRYPT 2011: 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings 30*. Springer, 2011, pp. 568–588.

[22] Y. Zhang, D. Zheng, X. Chen, J. Li, and H. Li, "Computationally efficient ciphertext-policy attribute-based encryption with constant-size ciphertexts," in *Provable Security: 8th International Conference, ProvSec 2014, Hong Kong, China, October 9-10, 2014. Proceedings 8*. Springer, 2014, pp. 259–273.

[23] S.-F. Sun, C. Zuo, J. K. Liu, A. Sakzad, R. Steinfeld, T. H. Yuen, X. Yuan, and D. Gu, "Non-Interactive Multi-Client Searchable Encryption: Realization and Implementation," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 1, pp. 452–467, Jan. 2022.