


Article

MDS² – C³PF: A Medical Data Sharing Scheme with Cloud-Chain Cooperation and Policy Fusion in IoT

Heng Pan ^{1,2,3,*} , Yaoyao Zhang ^{1,2,3,*} , Xueming Si ^{1,2,4}, Zhongyuan Yao ^{1,2,3} and Liang Zhao ⁵

¹ Research Institute of Advanced Information Technology, Zhongyuan University of Technology, Zhengzhou 450007, China

² Henan International Joint Laboratory of Blockchain and Data Sharing, Zhengzhou 450007, China

³ Henan Key Laboratory of Network Cryptography Technology, Zhengzhou 450007, China

⁴ Department of Computer Science and Engineer, Shanghai Jiao Tong University, Shanghai 200240, China

⁵ Department of Computing and Mathematics, Faculty of Philosophy, Science and Letters at Ribeirao Preto (FFCLRP) Bandeirantes, University of Sao Paulo, 3900, Ribeirao Preto 14040-901, SP, Brazil

* Correspondence: panheng@zut.edu.cn (H.P.); 2020016552@zut.edu.cn (Y.Z.)

Abstract: The Internet of Things (IoT) and cloud technologies have significantly facilitated healthcare. In such a context, medical data are collected by the terminals from the patients, manipulated, and stored on the cloud by hospitals (doctors). This brings asymmetry problems in medical data access control, processing, and storage between doctors and patients, which results in medical data sharing face many challenges such as privacy leakage and malicious feedback from cloud servers on queries. To solve these asymmetry problems, this paper proposes a medical data sharing scheme with cloud-chain cooperation and policy fusion in the IoT. Regarding asymmetrical access control rights, a conflict resolution and fusion algorithm that enables co-authorization of medical data by the doctor and the patient is introduced. To balance the symmetry of medical data storage and processing, a cloud-chain cooperation ciphertext retrieval method is proposed by means of two-stage joint searching from cloud servers and the blockchain, which can not only detect malicious medical data feedback from cloud servers, but also improve the data search efficiency. The security analysis showed that this scheme satisfies the confidentiality and verifiability of the retrieved information, and the feasibility of the proposed scheme was demonstrated through experiments.

Keywords: cooperation retrieval; co-authorization; policy conflict resolution; blockchain; IoMT



Citation: Pan, H.; Zhang, Y.; Si, X.; Yao, Z.; Zhao, L. MDS² – C³PF: A Medical Data Sharing Scheme with Cloud-Chain Cooperation and Policy Fusion in IoT. *Symmetry* **2022**, *14*, 2479. <https://doi.org/10.3390/sym14122479>

Academic Editors: Kuo-Hui Yeh and Liangmin Wang

Received: 24 October 2022

Accepted: 17 November 2022

Published: 23 November 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The massive amount of data generated by the Internet of Medical Things (IoMT) serves as an important vehicle for recording patient information for treatment and needs to be shared to advance medical information [1,2]. Because of the limited computational power and storage capacity of the terminal, medical data based on the IoMT are generally stored and processed on the cloud, where the data access and usage rights are always in the hands of medical institutions. This brings asymmetries in medical data access control, processing, and storage between doctors and patients. Specifically, patients have little control over their records, which results in privacy disclosure. Furthermore, the unbalanced structure of the terminal collection, cloud storage, and processing may cause malicious tampering or false feedback on queries from the cloud server. These problems due to asymmetry seriously impede medical data sharing.

Current research on IoMT data sharing mainly focuses on the access control of medical data and secure retrieval methods. The common method is to use Attribute-Based Encryption (ABE) [3], especially Ciphertext Policy Attribute-Based Encryption (CP-ABE) [4], access control to encrypt medical data and store them on the cloud server. Most studies based on CP-ABE to solve medical data sharing consider the access control of medical

data by hospitals or patients [5–7]. However, medical data come from both patients and doctors. Thus, their ownership should be shared between hospitals and patients. Therefore, it is worthwhile to further study the medical data access control method with the co-authorization of doctors and patients.

Furthermore, the ciphertext retrieval of medical data is a hot research topic that addresses the security risks of the cloud. While Searchable Encryption (SE) [8] can prevent the leakage and tampering of medical records by semi-trusted cloud servers, there still remains the risk of the cloud server's malicious feedback errors or false medical data [9,10]. Since these solutions do not break the centralized imbalanced structure, it is difficult to fundamentally solve the secure storage and processing problems of medical data sharing. Blockchain is open, transparent, tamperproof, and traceable, which can provide a solution to this problem [11]. The existing methods mainly use blockchain to record the query information [12–14] or ensure the correctness of the retrieved results by performing ciphertext medical data searching on the blockchain [15]. However, due to blockchain's special structure and storage limitations, the blockchain ciphertext retrieval efficiency is much lower than cloud services. Although blockchain breaks the asymmetric structure of cloud-based medical data's centralized storage and processing, it raises efficiency problems and other issues.

To solve the above two asymmetric problems that impede medical data sharing, a Medical Data Sharing Scheme with Cloud-Chain Cooperation and Policy Fusion in the IoT ($MDS^2 - C^3PF$) is proposed. The scheme makes use of cloud-chain cooperation to effectively balance the medical data sharing access control between doctors and patients and resist some of the security risks such as malicious feedback and privacy disclosure caused by the centralized asymmetric structure of cloud data processing.

The main contributions of this article are as follows:

- A multi-stage system model is proposed. The access control right of medical data between doctors and patients becomes symmetric through their co-authorization. A symmetric cloud-chain cooperation storage and retrieval method is designed to detect malicious feedback from the cloud and to improve the medical data retrieval efficiency.
- An attribute-based access policy fusion method is proposed to develop an access control policy created by both doctors and patients. When the medical data access control policies made by doctors and patients conflict, the balance score matrix is calculated to solve this by using the mutual influence weight and intention score of doctors and patients.
- Considering both medical records retrieval efficiency and detecting malicious feedback from cloud servers, a cloud-chain cooperation retrieval method is proposed. It can improve medical records retrieval efficiency by designing the off-chain search structure and performing an initial search on the cloud server with a secondary search on the blockchain.

The rest of this article is arranged as follows. Related work is first discussed in Section 2, followed by Section 3, which presents the background knowledge. Section 4 introduces the $MDS^2 - C^3PF$ model and the security model. The construction and details of the scheme are described in Section 5. Then, Section 6 discusses the security analysis. In Section 7, the experimental analysis of the scheme is carried out. Finally, the whole article is concluded.

2. Related Work

In order to ensure the security of medical data, many works encrypt medical records and store the ciphertext on the cloud. Many IoT-based medical data sharing researchers use ABE and SE to protect data security and privacy in cloud searching and access. In order to overcome the problems caused by the centralized nature of the cloud, recent studies have tried to use blockchain to make an improvement.

Since data sharing requires fine-grained access control methods, Sahai et al. [3] first presented an ABE scheme enabling one-to-many encryption. To improve the performance

of ABE, Bethencourt et al. [4] provided a CP-ABE method, which was proven secure in the generic group model. Based on these fundamental works, the state-of-the-art studies on secure data sharing on the cloud, especially medical data sharing, often make use of CP-ABE. Han et al. [16] proposed an attribute-revocable CP-ABE scheme based on privacy protection, which can share data securely by the cloud. In IoT research, Li et al. [17] presented a white-box traceable CP-ABE scheme with accountability in the IoT to address the user key abuse problem. Hu et al. [18] proposed a strategy-hidden sharing method in the IoT to outsource data to the cloud, which can reduce the cost of the user and improve computational efficiency. K. et al. [19] introduced a lightweight key management mechanism based on the IoT to solve the key escrow problem. At the same time, the development of the cloud and the IoT have greatly promoted technological innovation in the medical scene and promoted the secure sharing of medical data. In particular, in order to prevent the disclosure of patients' privacy, many existing ABE access control studies focus on how to strengthen patients' control over medical data. Hwang et al. [20] believed that patients have ownership of medical data and used CP-ABE to encrypt medical data to protect the privacy of patient data. Liu et al. [21] proposed an approach based on consortium blockchain to make access control policies by patients. Wang et al. [22] provided a fast, secure patient-controlled access scheme for medical data, which can reduce the storage capacity on the mobile terminal. In fact, the medical data are produced by both the patient and the doctor. Therefore, access to medical data should be decided jointly by doctors and patients, but there are not many studies on co-authorization.

Searchable encryption based on ABE is crucial to achieving secure data sharing. Song et al. [8] first introduced the scheme of searchable encryption and solved the problem of ciphertext retrieval. Since this pioneering work, many security research works have been proposed to improve the efficiency of ciphertext search and improve the search function. Li et al. [23] provided a secure and efficient dynamic searchable encryption scheme on medical cloud data, improving the ciphertext keyword search efficiency. Chen et al. [24] realized an efficient fuzzy search of keywords by encrypting the fuzzy association scores between data and query predicates. Chaudhari et al. [25] proposed a searchable encryption algorithm based on a single keyword that allows a user to access a subset of the documents. Tahir et al. [26] exploited the properties of the modular inverse to generate a probabilistic trapdoor that facilitates the search over the secure inverted index table. Sun et al. [27] proposed an attribute-based searchable encryption scheme that supports multiple data owners and data requesters. Zheng et al. [28] proposed a verifiable-attribute-based keyword search scheme that could prevent false feedback from the cloud. Yu et al. [29] retrieved the required ciphertext medical data in the IoT, reducing the computational load of outsourcing decryption and improving efficiency.

However, semi-trusted cloud servers are vulnerable to providing false feedback and the malicious forging of medical data. Blockchain's immutable and decentralized characteristics provide new research ideas and solutions. Liu et al. [30] implemented an electronic medical record sharing scheme based on policy hiding, which uses blockchain to store electronic medical record ciphertext and ensure the correctness of data retrieval. Cao et al. [31] presented a cloud-assisted secure medical system that uses blockchain to record the data operation process and ensure the traceability of data. Zhang et al. [32] provided a decentralized personal health record sharing scheme, using blockchain for the keyword search to ensure the correctness of the queried results. Krishna et al. [33] used ciphertext indexing to search data and utilized blockchain to verify every transaction to make medical data transmission more reliable. Zhu et al. [34] proposed a shared electronic medical data system, which used the automatic execution of chain codes to ensure data access security. In addition, there are some literature works on smart contracts in blockchain, which realize secure sharing and retrieval. Saini et al. [35] designed an access control framework based on a smart contract to prevent a single point of failure and ensure data sharing among different entities. Chen et al. [36] ensured the security of medical

data through smart authorization contracts. However, the retrieval efficiency of the above research schemes still needs to be improved.

In a word, it can be seen that the existing works mainly focus on the security data access and retrieval in the IoT and cloud environments, while the essential problem of asymmetry in medical data sharing is not discussed. Unlike the above methods, $MDS^2 - C^3PF$ solves the asymmetric access control right of the medical data between doctors and patients and the asymmetric collection and processing capability between the IoT and the cloud. Meanwhile, our scheme can detect false feedback from the cloud server and improve the sharing data retrieval efficiency.

3. Preliminaries

This section sorts out the preliminary knowledge, including the bilinear maps and access structure.

3.1. Bilinear Maps

Let G_1, G_2 be a multiplicative cyclic group of prime order p [37]; the generating element of G_1 is g . The bilinear map $e : G_1 \times G_1 \rightarrow G_2$ has the following characteristics:

- Bilinear: $\forall u, v \in G_1$ and $a, b \in \mathbb{Z}_p^*$ with $e(u^a, v^b) = e(u, v)^{ab}$;
- Non-degeneracy: $e(g, g) \neq 1$;
- Computability: $\forall u, v \in G_1$ with $e(u, v)$ computable.

3.2. Access Structure

Let T be an access control structure tree whose root node is r [38]. Each non-leaf node in the tree is a threshold, and the leaf nodes are attribute values. Let the number of child nodes of node x be num_x and k_x be a threshold ($0 \leq k_x \leq num_x$). If and only if at least k_x child nodes meet the condition, the parent node can obtain the correct result. When $k_x < num_x$, the current threshold gate is OR; when $k_x = num_x$, the current threshold gate is AND.

Let T_x be a subtree of tree T , where x is a child node of T :

- If there exists an attribute set S satisfying the access control tree T , then $T_r(S) = 1$;
- If x is a leaf node, if and only if the attribute set S contains the attribute $att(x) \in S$ of the current leaf node, then $T_x(S) = 1$;
- If x is a non-leaf node, for a child node x' of node x , compute $T_{x'}(S)$; if and only if at least k_x children return $T_{x'}(S) = 1$, it can be denoted as $T_x(S) = 1$.

4. System and Security Model

This section introduces the system model, algorithm definition, and security model.

4.1. System Model

Figure 1 shows the system model, which involves the following roles:

- Trust Center (TC). The TC generates key pairs for all legitimate users and executes the policy fusion algorithm.
- Blockchain (BC). The BC is a consortium blockchain that consists of multiple medical institutions to store index information.
- Doctor (DOC). The DOC is the medical data owner responsible for encrypting medical data and uploading the encrypted data to the cloud.
- Cloud Server (CS). The CS is responsible for storing the ciphertext of medical data and sending the file storage address to the DOC.
- Patient (PA). The PA is the owner of the medical data, responsible for developing access policies for the medical data.
- Data Requester (DR). The DR generates a search trapdoor to obtain the corresponding type of data from the cloud and decrypts the medical data.

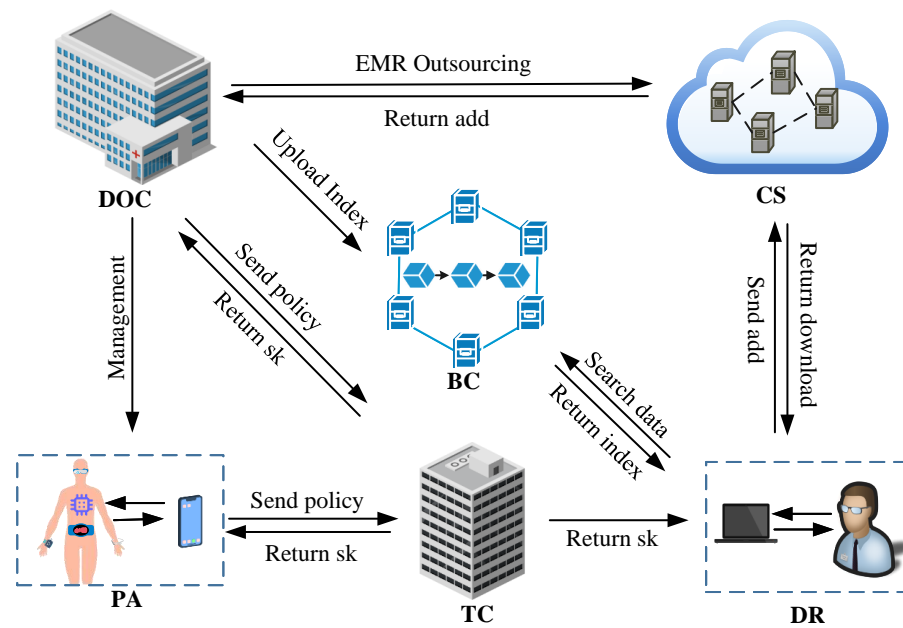


Figure 1. System model.

Table 1 shows the meanings of the symbols in our scheme.

Table 1. Symbols' description.

Symbols	Description
pp	Public parameter
pk	System public key
mk	Master key
M	Medical data
C_M	Medical data ciphertext
sk	Private key
H	Medical data ciphertext hash
k	Encryption key for medical data
P	Balance score matrix
Y	Impact score matrix
X	Intention score matrix
W	Keyword set
T	Access control policy tree
add	Storage Address

4.2. Scheme Definition

The proposed scheme consists of the following polynomial-time algorithms:

- $ParaSet(\lambda) \rightarrow (pp, pk, mk)$: Inputs security parameter λ and outputs public parameter pp and master key mk .
- $KeyGen(S, mk, pp) \rightarrow sk$: The TC inputs attribute set S , master key mk , and public parameter pp . Then, the TC outputs the attribute key sk for all legitimate users.
- $StrategyFus(A, B) \rightarrow T$: The TC inputs the access control policy tree A of the PA and the tree B of the DOC. The TC fuses A and B and expresses the result as T .
- $Enc(k, M) \rightarrow C_M$: The DOC inputs the encryption key k and the medical data and outputs the ciphertext C_M of the medical data.
- $IndexGen(W, T, v_1, k, mk, pk, pp, v_2) \rightarrow (I, C^*)$: The DOC inputs the keyword W , access control policy tree T , system public key pk , master key mk , and public parameter pp . Then, the DOC outputs index I and partial index ciphertext C^* .
- $TrapGen(W', mk, sk) \rightarrow Trap$: The DR runs the algorithm to generate trapdoors based on the keyword W' that needs to be queried and then uploads the trapdoors to the BC.

- $CloudSearch(Dep, Trap, sk) \rightarrow DATA^*$: The CS inputs medical type Dep , trapdoor $Trap$, and secret key sk . Then, the CS outputs the dataset $DATA^*$.
- $BlockSearch(Trap, DATA^*, I) \rightarrow Tx$: The BC executes the algorithm and performs the blockchain keyword search operation based on the trapdoor $Trap$ and the initial filtered dataset $DATA^*$ and outputs the transactions Tx .
- $Verify(Tx) \rightarrow l$: The DR obtains the hash value Tx , verifies whether the ciphertext is modified, and outputs $l = 1$ if the hash value is consistent; otherwise, $l = 0$.
- $Decrypt(l, sk, I) \rightarrow k$: If the medical data verification is passed, the DR will decrypt the key k of the medical data according to its own attribute private key sk and I .

4.3. Security Model

$MDS^2 - C^3PF$ performs keyword retrieval on the blockchain to improve searching security and address the issue of false feedback from the cloud servers. Two security models are defined: the Indistinguishability of Ciphertext under the Selectively Chosen Keyword Attack (INDC-SCKA) and the Keyword Secrecy under the Selectively Chosen Secret Key Attack (KS-SCSKA).

4.3.1. The Definition of INDC-SCKA

Theorem 1. To prove the INDC-SCKA of $MDS^2 - C^3PF$, in this paper, let attacker A_1 and challenger B_1 play a secure game $Game_1$. $MDS^2 - C^3PF$ is said to be indistinguishable with keywords if the probability of attacker A_1 winning the game is negligible in polynomial time.

Initialization: B_1 inputs the security parameter λ and runs $ParaSet(\lambda)$. Finally, the initialized algorithm returns the system parameter pp and the master key mk .

Phase 1: A_1 initiates a trapdoor query on the keyword set W_1, \dots, W_t .

- $TrapGen(W, mk, sk)$: B_1 runs the trapdoor generation algorithm $Trap(W_m, mk, sk)$ to return the trapdoor T_{W_m} and then returns it to A_1 .

Challenge: A_1 sends the keyword set (W_0, W_1) to B_1 , where W_0, W_1 is of equal length. B_1 selects a bit $c \in \{0, 1\}$. B_1 generates $I_c = IndexGen(W_c, T', v_1, k, mk, pk, pp, v_2)$ and sends I_c to A_1 .

Phase 2: A_1 issues a trapdoor query for the keyword set W_{m+1}, \dots, W_τ .

- $Trap(W_i \neq W_0, W_1, mk, sk)$: B_1 runs $Trap(W_i, mk, sk)$ to obtain the trapdoor T_{W_i} , which is sent to A_1 .

Guess: A_1 outputs guess $c' \in \{0, 1\}$. If $c = c'$, A_1 wins the game.

The probability of success of A_1 attacking the model is $Adv_{A_1}(1^k) = |Pr[c = c'] - \frac{1}{2}|$.

4.3.2. The Definition of KS-SCSKA

Theorem 2. To prove the KS-SCSKA, a secure game $Game_2$ is defined between attacker A_1 and challenger B_1 . If the probability of A_1 completing the keyword secrecy game in polynomial time is negligible, then $MDS^2 - C^3PF$ can achieve the keyword security.

Initialization: B_1 inputs the security parameter λ and runs $ParaSet(\lambda)$. Finally, the initialized algorithm returns the system parameter pp and the master key mk .

Phase 1: A_1 interrogates the following algorithm in polynomial time.

- $KeyGen(S, mk, pp)$: B_1 first gives the key sk to A_1 and adds the key set to I_{KeyGen} .

- $TrapGen(W, mk, sk)$: Given sk and W , B_1 executes the trapdoor generation algorithm to generate the trapdoor. B_1 sends the result to A_1 .

Challenge: A_1 selects the challenge key sk and gives it to B_1 . B_1 selects the key set W' from the information space and executes the $IndexGen$ algorithm, and then, B_1 gives the index to A_1 .

Guess: After A_1 obtains a different set of keywords τ , the adversary outputs W' . If $W' = W$, then A_1 wins.

$MDS^2 - C^3PF$ can achieve keyword security if the probability of A_1 winning the game is no more than for $(|\mathcal{W}| - \tau)^{-1} + \epsilon$. τ denotes the number of keyword sets; ϵ is the negligible probability in the security parameter k ; \mathcal{W} is the keyword space.

5. Scheme Construction

Our scheme includes five stages, shown in Figure 2. In the first stage, the system initializes the parameters. Then, the TC sends private keys to the users. In order to form one consistent access policy, in Stage 2, a policy fusion algorithm is presented to merge and resolve conflicts between the access control policies proposed by the patient and the doctor, respectively. In the data generation and storage stage, a doctor uses the fused policy to encrypt and upload the medical data. The original ciphertexts are stored on the cloud, and the index information is stored on the blockchain. The following two stages leverage cloud-chain cooperation mapping and searching to implement controlled secure access to the medical data. The following subsections will thoroughly discuss the access control policy fusion algorithm and the detailed working process of the five stages.

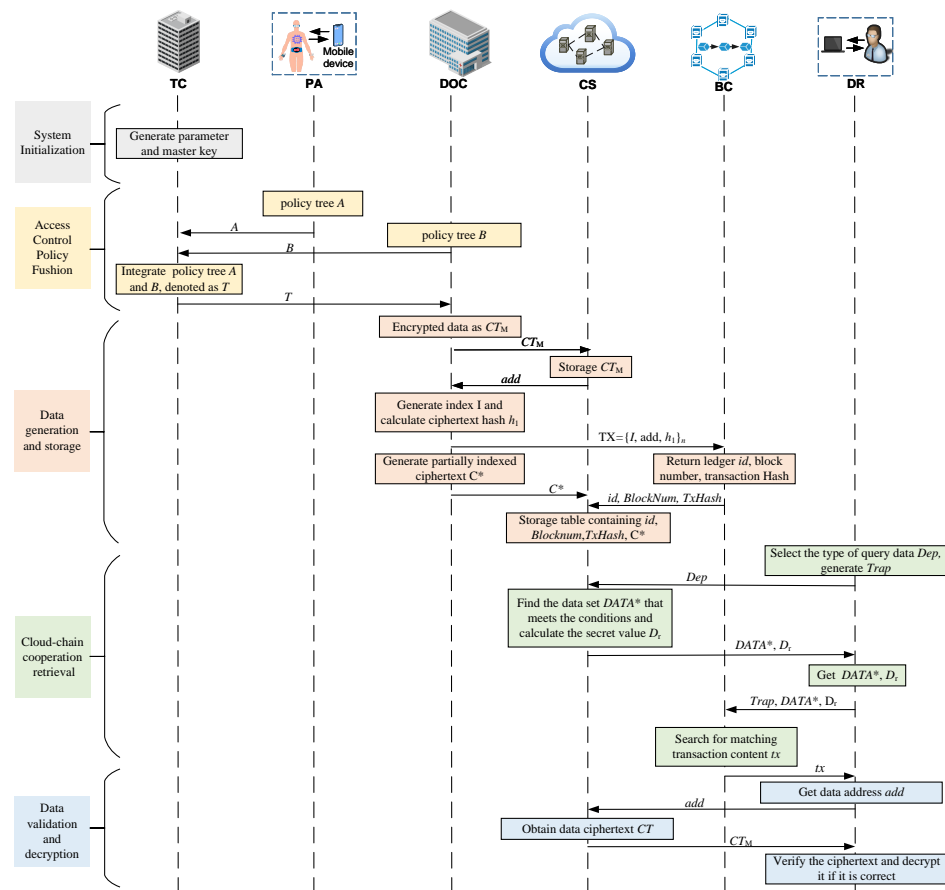


Figure 2. Scheme working process.

5.1. Access Control Policy Fusion Algorithm

The first step of our scheme is to create one authorization policy that considers User A and B’s access control over medical data. User A generates policy set Str_{A} , which contains n policies, where a_i is the policy of Str_{A} . Similarly, User B produces a policy set Str_{B} that contains m policies, where b_j is a policy of Str_{B} . The TC takes charge of generating a co-authorization policy set Str_{T} by comparing Str_{A} and Str_{B} . Firstly, the TC puts the same policies from setting Str_{A} and Str_{B} into the new policy set Str_{T} . Then, different and non-conflicting policies from Str_{A} and Str_{B} are also added to Str_{T} . The most critical work is to call the policy conflict resolution algorithm when there are conflicting policies in Str_{A} and Str_{B} .

Figure 3 shows the flowchart of the algorithm.

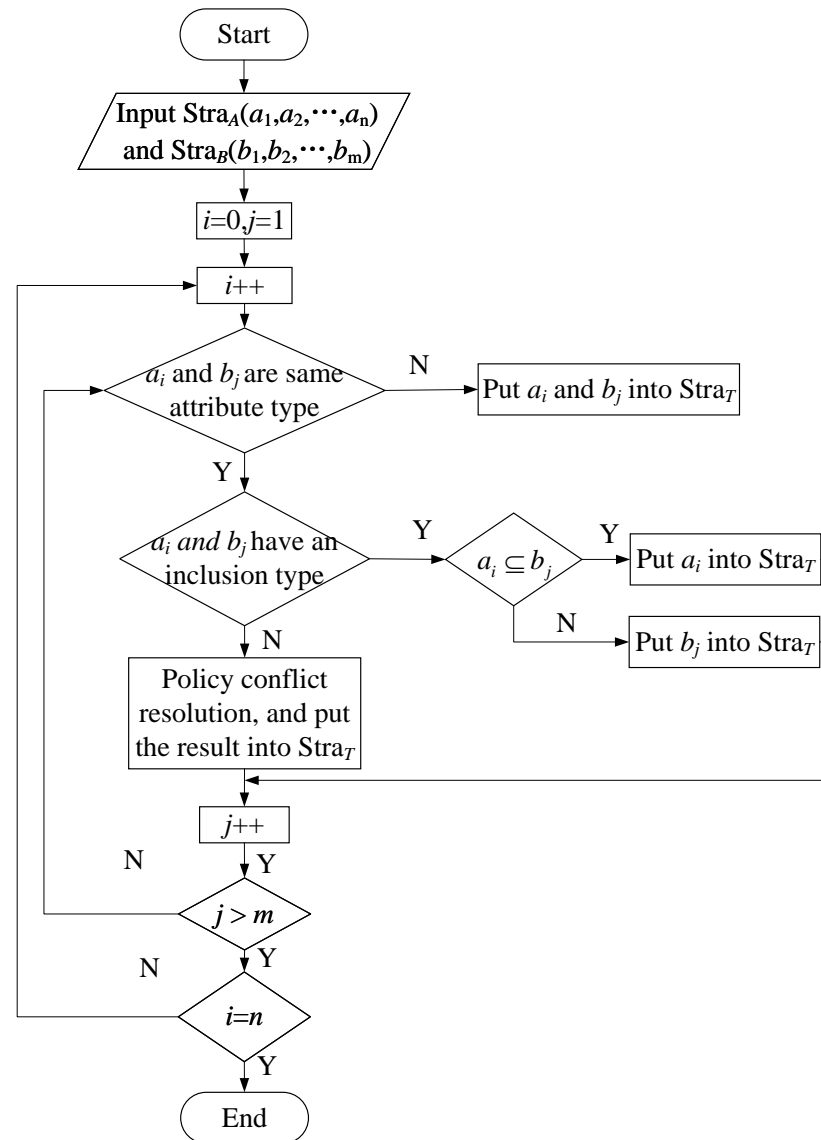


Figure 3. Policy fusion algorithm.

5.1.1. Policy Conflict Resolution Algorithm

The strategy conflict resolution is an improved algorithm presented by Tan et al. [28]. They indicated a peer-aware collaborative access control based on identity, which achieves policy equilibrium through peer influence. Differently, our method achieves attribute-based policy equilibrium, referring to players' influence and using the strategy to update the rules. In this paper, the intention score is defined as the user's willingness intensity score for policy a_i . The impact score is the intensity of user interaction. The balance score is the user's final willingness value for a policy after being influenced by others:

- Initialize the matrix:
Let it contain n users (resource owners) and f conflicting attributes, then initialize user i 's intention score for conflicting attribute a_k as $x_i(a_k)$, where $i \in (1, \dots, n)$, $k \in (1, \dots, f)$. The value range of the intention score is 0 to 5. Intention score matrix X is denoted as

$$X = \begin{bmatrix} x_1(a_1) & \cdots & x_1(a_f) \\ \vdots & \ddots & \vdots \\ x_n(a_1) & \cdots & x_n(a_f) \end{bmatrix} = [I_1 \quad \cdots \quad I_f] \tag{1}$$

Suppose the initialized user i is influenced by user j 's influence score w_{ij} , taking values in the range of 0–1. Impact score matrix Y is denoted as

$$Y = \begin{bmatrix} w_{11} & \cdots & w_{1n} \\ \vdots & \ddots & \vdots \\ w_{n1} & \cdots & w_{nn} \end{bmatrix} \tag{2}$$

- Generate the balance score matrix:
Let the sentiment gain of user $i \in U$ for an attribute a_k be

$$pay_i = x_i(a_k)p_i(a_k) - \frac{1}{2}(p_i(a_k))^2 + \sum_{j \neq i} w_{ij}p_i(a_k)x_j(a_k) \tag{3}$$

where $x_i(a_k)$ is the initial willingness value set by the user and $p_i(a_k)$ is the final willingness value influenced by other users.

$$\frac{dpay_i}{dp_i(a_k)} = x_i(a_k) - p_i(a_k) + \sum_{j \neq i} w_{ij}x_j(a_k) \tag{4}$$

Let $\frac{dpay_i}{dp_i(a_k)} = 0$, which gives

$$p_i(a_k) = x_i(a_k) + \sum_{j \neq i} w_{ij}x_j(a_k) \tag{5}$$

That is, when $p_i(a_k) = x_i(a_k) + \sum_{j \neq i} w_{ij}x_j(a_k)$, the user has the highest gain. From Equation (5), the final user's intention is the sum of his/her own and the player's influence score. If there is no player influence, then $p_i(a_k) = x_i(a_k)$. The final willingness value of each player is calculated by computing the user's initial settings. Therefore, the column vector P of the balance score matrix p_k is denoted as

$$\begin{aligned} p_k &= Y \cdot I_k = \begin{bmatrix} w_{11} & \cdots & w_{1n} \\ \vdots & \ddots & \vdots \\ w_{n1} & \cdots & w_{nn} \end{bmatrix} \cdot \begin{bmatrix} x_1(a_k) \\ \vdots \\ x_n(a_k) \end{bmatrix} \\ &= \begin{bmatrix} w_{11} \cdot x_1(a_k) + \cdots + w_{1n} \cdot x_n(a_k) \\ \vdots \\ w_{n1} \cdot x_1(a_k) + \cdots + w_{nn} \cdot x_n(a_k) \end{bmatrix} \end{aligned} \tag{6}$$

Let $p_i(a_k) = w_{i1} \cdot x_1(a_k) + \cdots + w_{in} \cdot x_i(a_k)$, and thus, P is denoted as

$$\begin{aligned} P &= [p_1 \quad \cdots \quad p_f] = \begin{bmatrix} q_1 \\ \vdots \\ q_n \end{bmatrix} \\ &= \begin{bmatrix} p_1(a_1) & \cdots & p_1(a_f) \\ \vdots & \ddots & \vdots \\ p_n(a_1) & \cdots & p_n(a_f) \end{bmatrix} \end{aligned} \tag{7}$$

- Judge rule:
This rule determines whether the policy is successfully merged. Firstly, it compares the size of each value of q_i in the row vector, selects the attribute a_k corresponding to

the largest value, and stores it in the attribute selection set γ . Next, the rule judges whether the attribute a_k in the set γ is the same attribute and outputs the result.

- Modify the intention matrix:
Calculate the probability that user j is referenced by other users.

$$Pro_{i \rightarrow j}(a_k) = 1 / \left(1 + e^{(p_i(a_k)) - p_j(a_k) / w_{ij}} \right) \quad (8)$$

where $Pro_{i \rightarrow j}(a_k)$ denotes the probability that i imitates j 's strategy under attribute a_k and w_{ij} denotes the fraction of i influenced by j . $p_i(a_k)$ and $p_j(a_k)$ denote the equilibrium fraction of users i and j choosing attribute a_k . For each user j , the average probability:

$$P_j(a_k) = \frac{\sum_{i=1, i \neq j}^n P_{i \rightarrow j}}{(n-1)} \quad (9)$$

where the user j with the highest probability is selected and all other users modify the intention score referring to the policy of j . The balance matrix P is recalculated to determine whether the users' choices are consistent.

5.1.2. Policy Conflict Resolution Algorithm Process

Compared with the mechanism proposed by Tan et al. [39], the policy conflict resolution algorithm can achieve finer-grained conflict resolution and is suitable for co-authorization scenarios such as the IoMT. Algorithm 1 shows the detailed policy fusion algorithm.

Algorithm 1 Policy conflict resolution algorithm.

Input: $X, Y, S^*, \gamma = []$

Output: $\gamma[i]$

```

1. while  $S^* \neq \text{null}$  do
2.    $S^* = S^* - \{a_k\}$ 
3.   compute  $p_i = Y \cdot I_k$ 
4. end while
5. while true do
6.   let  $P = [p_1 \ \dots \ p_f]$ 
7.    $result = \text{JudgeRule}(P)$ 
8.   if  $result = Y$  do
9.     break
10.  end if
11.  if  $result = N$  do
12.    for  $i = 1; i < f; i++$  do
13.       $user = \text{getMaxUser}(Y, P, f)$ 
14.      for  $j = 1; j < n; j++$  do
15.        if  $j \neq user$  do
16.          update  $x_j(a_f)$  according to  $x_{user}(a_f)$ 
17.        end if
18.      end for
19.    end for
20.    Compute  $p_i = Y \cdot I_k$ 
21.  end if
22. end while
23. return  $\gamma[i]$ 

```

Firstly, the TC initializes intention matrix $X = [I_1, \dots, I_f]$, impact matrix Y , and conflicting attribute set S^* . It initializes attribute selection set γ , which here is represented as an array $\gamma = []$. The TC calculates the balanced matrix P for the conflicting attributes and checks if all users select the same conflicting attribute. If yes, the algorithm ends.

Otherwise, the TC calculates the probability of the user being imitated and selects the user with the highest probability. Other users follow the highest-probability user to modify the intention matrix. After that, the balanced matrix P is recalculated. Figure 4 shows the specific process of the algorithm.

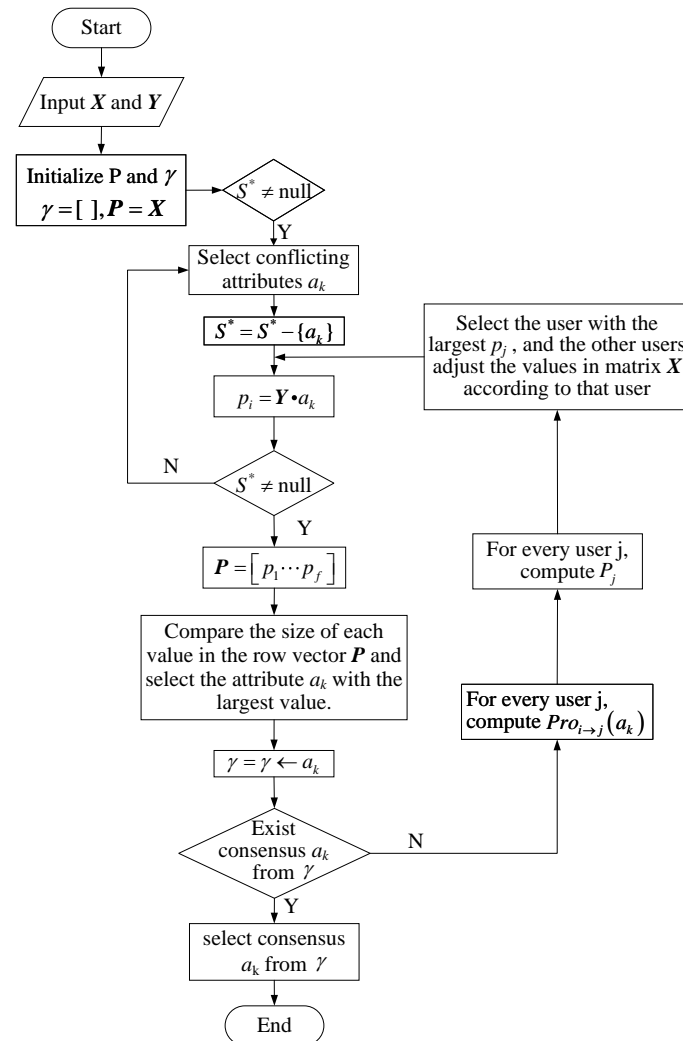


Figure 4. Policy conflict resolution algorithm.

5.2. Details of Five Stages

The project consists of five stages: system initialization, access control policy fusion, data generation and storage, cloud-chain cooperation retrieval, and data verification and decryption.

Stage 1. System initialization:

The TC initializes the parameter and generates the private key for the PA, DOC, and DR, respectively.

- System parameter setting:
Given security parameter λ and mapping parameter (G_1, G_2, q, g, e) , the TC executes $ParaSet(\lambda)$ to generate parameter pp , system public key pk , and master key mk . Then, it selects two hash functions $H_1\{0, 1\}^* \rightarrow G_1, H_2\{0, 1\}^* \rightarrow Z_q$. Besides, the TC chooses $\alpha, \beta, \gamma \in Z_q$ and computes $t_1 = g^\alpha, t_2 = g^\beta, t_3 = g^\gamma$. Finally, the TC generates public parameter $pp = (G_1, G_2, q, g, e, H_1, H_2)$, system public key $pk = (t_1, t_2, t_3)$, and master key $mk = (\alpha, \beta, \gamma)$.
- Key generation:

The TC executes $KeyGen(S, mk, pp)$ and generates key sk for the user who owns attribute set S . Firstly, the TC selects random $r \in Z_q$ and chooses $r_s \in Z_q$ for every attribute $s \in S$. Then, it computes $D_1 = g^{(\alpha\gamma-r)/\beta}$, $D_2 = g^{(\alpha+r)/\beta}$ and calculates $A_s = g^r H_1(s)^{r_s}$, $B_s = g^{r_s}$. Finally, it generates the user's attribute key $sk = (D_1, D_2, \{A_s, B_s\}_{s \in S})$.

Stage 2. Access control policy fusion:

In this stage, the TC is responsible for executing the access policy fusion algorithm to create a new policy set $stra_T$. Here, an example is given to show how the conflict of a policy is fused.

For the conflicting attributes a_1 and a_2 , the TC obtains the intention scores of the conflicting attributes of the DOC and PA, as shown in Table 2. Besides, the TC obtains the impact scores of the attributes, as expressed in Table 3. Then, the balance score matrix $P = [p_1 \ p_2]$ is calculated, where $p_1 = Y \cdot (x_1(a_1) \ x_2(a_1))^T$, $p_2 = Y \cdot (x_1(a_2) \ x_2(a_2))^T$, as shown in Table 4.

Table 2. Intention score matrix X .

	a_1	a_2
PA	$x_1(a_1)$	$x_1(a_2)$
DOC	$x_2(a_1)$	$x_2(a_2)$

Table 3. Impact score matrix Y .

	PA	DOC
PA	1	w_{12}
DOC	w_{21}	1

Table 4. Balance score matrix P .

	a_1	a_2
PA	$x_1(a_1) + x_2(a_1) \cdot w_{12}$	$x_1(a_2) + x_2(a_2) \cdot w_{12}$
DOC	$x_1(a_1) \cdot w_{21} + x_2(a_1)$	$x_1(a_2) \cdot w_{21} + x_2(a_2)$

The current user's choice is the highest score in the row vector of P . If the attributes corresponding to the highest scores of the DOC and PA are the same, this attribute is the final result. If there is no agreement on the attribute, the TC calculates the probability $P_{1 \rightarrow 2}(a_1) = 1 / (1 + e^{(p_1(a_1) - p_2(a_1)) / w_{12}})$ of the PA imitating the DOC's strategy under attribute a_1 . Similarly, the TC calculates the probability $P_{2 \rightarrow 1}(a_1)$ of the PA being imitated. Under attribute a_1 , the average probability of the PA being imitated is $P_1(a_1) = P_{2 \rightarrow 1}(a_1)$, and the average probability of the DOC being imitated is $P_2(a_1) = P_{1 \rightarrow 2}(a_1)$. If $P_1(a_1) > P_2(a_1)$, the DOC modifies the intention score to attribute a_1 according to the PA. Furthermore, the TC calculates $P_1(a_2)$ and $P_2(a_2)$. If $P_1(a_2) < P_2(a_2)$, the PA modified the intention score to attribute a_2 according to the DOC. Finally, the balance score matrix P is recalculated, and the algorithm ends when the highest score of row vectors in P is the same attribute.

Stage 3. Data generation and storage:

The DOC first encrypts the medical data with a symmetric encryption algorithm to obtain the ciphertext of the medical data and sends it to the CS for storage. Next, he/she encrypts the keyword set with policy tree T to generate the keyword index. Finally, he/she uploads the index information to the blockchain and stores the transaction information table on the cloud:

- Encryption of medical data:
The DOC inputs the medical data M and randomly selects a symmetric key k from the key space, then outputs C_M . The DOC stores C_M to the CS and obtains the storage address add . Moreover, the DOC performs a hash operation for ciphertext C_M to

obtain the result $h_1 = h(C_M)$, which ensures that the medical data on the cloud are neither tampered with nor forged.

- **Index generation:**
The DOC selects random $v_1, v_2 \in Z_q$ and generates an access control policy tree T with v_2 as the secret value. Then, the DOC encrypts the keyword set W . The specific algorithm is shown below:
 - (1) The DOC computes $C_0 = t_1^{v_2}$, $C_1 = t_2^{v_2}$, $C_2 = t_3^{v_1}$, and $K = ke(g, g)^{\alpha v_2}$.
 - (2) For each keyword $m \in \{1, \dots, t\}$, the DOC computes $\{\rho_m = t_1^{v_1 H_2(\omega_m)}\}_{m \in \{1, \dots, t\}}$.
 - (3) For each leaf node $z \in Z$, the DOC computes $\rho_z = g^{qz(0)}$, $\zeta_z = H_1(att(z))^{qz(0)}$.
 - (4) Finally, $I = (C_0, C_1, C_2, K)$, $C^* = \{\rho_m\}_{m \in \{1, \dots, t\}}, \{\rho_z, \zeta_z\}_{z \in Z}$.
- **Data storage:**
The DOC puts $n \{I, add, H\}$ in a transaction sheet $TX = \{I, add, h_1\}_n$. If the number of correct node verification results is more than $2/3$, the transaction is uploaded to the blockchain. The system obtains the transaction information from the blockchain, constructs the cloud-chain cooperation mapping table according to Dep , and stores it on the cloud, as shown in Table 5.

Table 5. Cloud-chain cooperation mapping table.

Department	Ledger ID	Block Number	Transaction Hash	Partial Ciphertext Index
<i>Dep</i>	<i>id</i>	<i>Blocknum</i>	<i>TxHash</i>	<i>C*</i>

Stage 4. Cloud-chain cooperation retrieval:

When the DR wants to obtain medical data, he/she generates a trapdoor by sk and sends it to the cloud server for retrieval. Then, the DR uploads the matching dataset to the consortium blockchain for secondary retrieval and performs a ciphertext keyword search operation. The specific process is as follows:

- **Trapdoor generation:**
 - (1) The DR selects random $p \in Z_q$ and chooses the keyword set $W' = \{w_1', \dots, w_m'\}$ ($m \in \{1, \dots, t\}$).
 - (2) The DR calculates $R_1 = \prod_{m=1}^t g^{p\alpha H_2(w'_m)}$, $R_2 = g^{p\gamma}$, $R_3 = D_1^p$, $A_s' = A_s^p$, and $B_s' = B_s^p$, where $s \in S$.
 - (3) The DR generates the trapdoor $Trap = (S, R_1, R_2, R_3, \{A_s', B_s'\}_{i \in S})$.
- **Cloud search:**
The CS runs $CloudSearch(Dep, Trap, sk)$. According to the medical data type Dep , the CS finds the corresponding medical dataset $DATA$ from the cloud-chain cooperation mapping table and then performs access control policy matching to select the dataset $DATA^*$ that satisfies the conditions.
 - (1) If x is a leaf node, let $i' = att(x)$, then the CS calculates $D_x = \frac{e(A_{i'}', \rho_x)}{e(B_{i'}', \zeta_x)} = e(g, g)^{r p q x(0)}$.
 - (2) If x is a non-leaf node, let x' be a child node of x . The CS computes if $D_x = \prod_{x' \in \omega_x} D_{x'}^{\Delta_{i, \omega_{x'}}(0)} = e(g, g)^{r p q x(0)}$ holds, where $i = index(x')$. If not, $D_{x'} = \perp$.
 - (3) Let $data = (id, Blocknum, TxHash)$, $D_r = e(g, g)^{r p q r(0)} = e(g, g)^{r p v_2}$.
 - (4) Finally, the CS outputs the dataset $DATA^* = \{data, D_r\}_d, d \in D$.
- **Blockchain search:**
The BC executes $BlockSearch(Trap, DATA^*, I)$. According to the trapdoor uploaded by the DR, the nodes on the blockchain carry out keyword matching by Equation (10).

$$e\left(\prod_{m=1}^t \rho_m C_0, R_2\right) = e(C_2, R_1) D_r e(C_1, R_3) \quad (10)$$

If Equation (10) holds, it means that the relevant data are queried and index I is returned; otherwise, \perp is returned. The correctness of Equation (10) is verified as follows:

$$\begin{aligned}
& e(C_2, R_1) D_r e(C_1, R_3) \\
&= e(t_3^{v_1}, \prod_{i=1}^t g^{p\alpha H_2(w'_{m'})}) e(g, g)^{r p v_2} e(t_1^{v_2}, D^p) \\
&= e(g^{y v_1}, \prod_{i=1}^t g^{p\alpha H_2(w'_{m'})}) e(g, g)^{r p v_2} e(g^{\beta v_2}, g^{(\alpha\gamma-r)\cdot p/\beta}) \\
&= e(g, g)^{y v_1 p \alpha \sum_{i=1}^t H_2(w'_{m'})} \cdot e(g, g)^{r p v_2} \cdot e(g, g)^{v_2 p (\alpha\gamma-r)} \\
&= e(g, g)^{p\alpha\gamma(v_2+v_1 \sum_{i=1}^t H_2(w'_{m'}))} \\
&= e(\prod_{m=1}^t \rho_m C_0, R_2)
\end{aligned} \tag{11}$$

Stage 5. Data verification and decryption:

- **Medical data validation:**
After successful retrieval, the DR obtains the medical data ciphertext CT_M from the CS. The DR executes $Verify(Tx)$ to verify the hash value. The DR calculates $h_2 = h(CT_M)$. If $h_2 = h_1$, it outputs $l = 1$; otherwise, it returns $l = 0$.
- **Ciphertext decryption:**
If $l = 1$, the DR calculates

$$k = \frac{K}{((e(C_1, D_2))^p / D_x)^{\frac{1}{p}}} \tag{12}$$

If Equation (12) holds, the DR can recover k by $Decrypt(l, sk, I)$ and further recover plaintext data M .

The correctness of Equation (12) is verified as follows:

$$\begin{aligned}
& \frac{K}{((e(C_1, D_2))^p / D_x)^{\frac{1}{p}}} \\
&= \frac{ke(g, g)^{\alpha v_2}}{\frac{e(g, g)^{(\alpha+r)v_2 p}}{e(g, g)^{r p v_2}}} \\
&= \frac{ke(g, g)^{\alpha v_2}}{\left(\frac{e(g, g)^{\alpha v_2 p} e(g, g)^{r v_2 p}}{e(g, g)^{r p v_2}}\right)^{\frac{1}{p}}} \\
&= \frac{ke(g, g)^{\alpha v_2}}{e(g, g)^{\alpha v_2}} \\
&= k
\end{aligned} \tag{13}$$

6. Security Analysis

We provide two detailed safety analyses of the proposed scheme, including the indistinguishability of ciphertext under the selectively chosen keywords attack and the keyword secrecy under the selectively chosen secret key attack.

6.1. The Security Analysis of Our Scheme under the INDC-SCKA

Theorem 3. *This scheme is selection-safe under the adaptive selection keyword attack based on the general bilinear group model.*

Proof. The challenger first establishes the general bilinear group model assumption. Let the hash functions H_1, H_2 be one-way hash functions, and the specific challenge process is as follows.

Initialization: B_1 selects $\alpha, \beta, \gamma \in Z_q$ and generates public parameter pp for A_1 and system public key $pk = (t_1, t_2, t_3)$. A_1 selects policy tree T' and returns it to B_1 . $H_1(i)$ simulates: If the attribute s has been queried, then challenger B_1 selects $r'_s \in Z_q$, inputs (s, r'_s) into O_{H_1} , and returns $g^{r'_s}$. Otherwise, challenger B_1 retrieves r'_s from O_{H_1} and returns $g^{r'_s}$.

Phase 1: Adversary A_1 asked O_{KeyGen} and O_{Trap} as follows:

- $O_{KeyGen}(S, mk, pp)$: Challenger B_1 chooses $r^* \in Z_q$ and calculates $D_1 = g^{(\alpha\gamma - r^*)/\beta}$, then randomly selects $r_s^* \in Z_q$ and calculates $A_s = g^{r_s^*} H_1(i)^{r_s^*}$, $B_s = g^{r_s^*}$ for the attribute $s \in S$. Finally, B_1 returns $(D_1, D_2, \{A_s, B_s\}_{s \in S})$.

- $O_{Trapdoor}(sk, W^*, mk)$: Challenger B_1 interrogates $KeyGen(S, mk, pp)$ to obtain $sk = (S, D_1, D_2, \{A_s, B_s\}_{s \in S})$. After that, B_1 randomly selects $s \in Z_q$ and computes $R_1 = \prod_{i=1}^t g^{sA_i H_2(w_i)}$, $R_2 = g^{s\gamma}$, $R_3 = D_1^s$. If S satisfies T' , W^* is added to the keyword set list L_W .

Challenge phase: Letting W_0, W_1 that does not belong to L_W , B_1 chooses $v_1, v_2 \in Z_q$, $v \in Z_q$ and calculates the secret value for each leaf node in T' . After that, B_1 selects random element $b^* \in \{0, 1\}$. If $b^* = 0$, B_1 outputs $\{\rho_i = g^{vH_2(w_m)}\}_{m \in \{1, \dots, t\}}$, $C_0 = t_1^{v_2}$, $C_1 = t_2^{v_2}$, $\{\rho_y = g^{qz(0)}, \zeta_y = H_1(att(z)^{qz(0)})\}_{z \in \ln}$, $C_2 = t_3^{v_1}$; otherwise, B_1 returns $\{\rho_m = t_1^{v_2 H_2(w_m)}\}_{m \in \{1, \dots, t\}}$, $\{\rho_z = g^{qz(0)}, \zeta_z = H_1(att(z)^{qz(0)})\}_{z \in \ln}$, $C_0 = t_1^{v_2}$, $C_1 = t_2^{v_2}$, $C_2 = t_3^{v_1}$.

Phase 2: Generate queries on the key generation algorithm and trapdoor algorithm as in Phase 1.

Guess: A_1 outputs guess $c' \in \{0, 1\}$. If $c = c'$, A_1 will succeed in the challenge and outputs 1; otherwise, A_1 fails in this challenge.

Analysis: If A_1 can construct $t_t^{\psi v_1 H_2(w_i)}$ for g^ψ contained in the output of the data that have been queried, A_1 is able to distinguish between $t_t^{v_1 H_2(w_i)}$ and g^v . Therefore, it is necessary to prove that A_1 can construct $e(g, g)^{\psi \alpha v_1 H_2(w_i)}$ from g^ψ with negligible probability. Because v_1 is only in γv_1 , let $\psi = \psi' \gamma$. A_1 only needs to construct $e(g, g)^{\psi' \gamma \alpha v_1}$ through the term γv_1 . When $\beta v_2 (\alpha \gamma - r^*) / \beta = v_2 (\alpha \gamma - r^*)$, A_1 needs to eliminate $v_2 r^*$ by r^* and $q_r(0)$. However, it is difficult to construct $v_2 r^*$. Therefore, A_1 needs to have properties that satisfy the access control tree T' to return the correct indexed result. Therefore, A_1 can win with negligible probability. Finally, the INDC-SCKA is implemented, which can effectively detect malicious feedback from the cloud server. \square

6.2. The Security Analysis of Our Scheme under the KS-SCSKA

Theorem 4. When a one-way hash function H_2 is given, this method is the keyword secrecy under selectively chosen secret key attack.

Proof. B_1 plays the following KS-SCSKA game.

Initialization: B_1 chooses $\alpha, \beta, \gamma \in Z_q$, selects hash function $H_1 : \{0, 1\}^* \rightarrow Z_q$, and finally, generates public keys $pk = (g^\alpha, g^\beta)$, $pp = (e, g, q)$, $mk = (\alpha, \beta, \gamma)$.

B_1 simulates $O_{H_1}(s)$ as follows: If the attribute s has not been queried before, B_1 randomly selects element $r_s \in Z_q$, adds (s, r_s) to O_{H_1} , and outputs g^{r_s} . Otherwise, B_1 retrieves r_s from O_{H_1} and returns g^{r_s} .

Phase 1: A_1 adaptively interrogates O_{KeyGen} and $O_{Trapdoor}$ in polynomial time:

- $O_{KeyGen}(S, mk, pp)$: B_1 interrogates the key generation algorithm and returns sk to A_1 , then adds the access policy T to the list L_{KeyGen} .

- $O_{Trapdoor}(sk, W^*, mk)$: Challenger B_1 first interrogates the trapdoor generation algorithm to obtain $sk = (T, \{A_z, B_z | z \in \ln(T)\})$ and then interrogates the trapdoor algorithm to return the trapdoor to adversary A_1 .

Challenge phase: A_1 selects the attribute set S' , B_1 , then selects T' to interrogate $KeyGen$ to obtain sk . Given S' and sk' , A_1 randomly selects W' to compute the ciphertexts CT'_M and trapdoor.

Guess: A_1 outputs keyword W' and sends it to B_1 , then B_1 asks the IndexGen algorithm to obtain index I' . If the keywords ciphertext is searched, A_1 wins the game.

Analysis: Since $|\mathcal{W}| - \tau$ is the space size of the remaining keyword set, the probability of A_1 computing W' from $H_2(W')$ is negligible. If A_1 queries τ different keywords, the maximum probability of winning the game is $(|\mathcal{W}| - \tau)^{-1} + \epsilon$. Therefore, it is proven that this scheme can achieve keyword secrecy. \square

7. Experiments and Performance Analysis

The experiment was implemented on the Ubuntu operating system and Intel Core i5 processor 2.3GHz. Fabric was used to set up a 4-node blockchain and Caliper to perform the stress test.

7.1. Blockchain Performance Analysis

As shown in Figure 5, the blockchain throughput was tested by increasing the transaction number from 1000 to 12,000 and using medical data index information to form a transaction. From the figure, it can be seen that the throughput is proportional to the number of transactions, and the change is relatively smooth. This indicates that the system performance improves steadily with the increase of transactions and has good scalability. Since upload indicates writing data to the blockchain, the uploading throughput is lower than downloading. The experiment shows that the proposed scheme in this paper can support data transactions in large quantities.

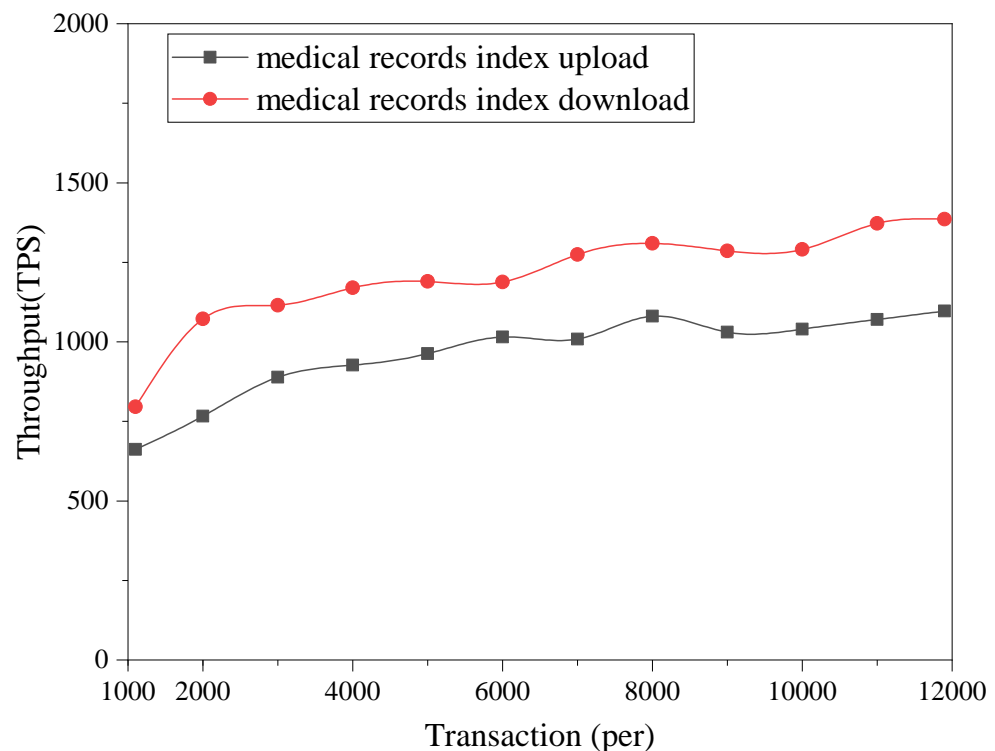


Figure 5. Consortium blockchain throughput testing.

Figure 6 shows the time comparison between the blockchain search and the cloud-chain combined search. As shown in the figure, a medical data requester should first obtain the dataset that satisfies the access policy from the cloud and then perform a secondary search on the blockchain. During access policy matching, the number of attributes in the user's private key affects the policy execution time, eventually affecting the search time. As illustrated, the cloud-chain searching scheme is better than blockchain searching. Because centralized cloud searching is more efficient than on-chain searching, our solution

combines them together to ensure search efficiency while preventing centralized cloud false feedback.

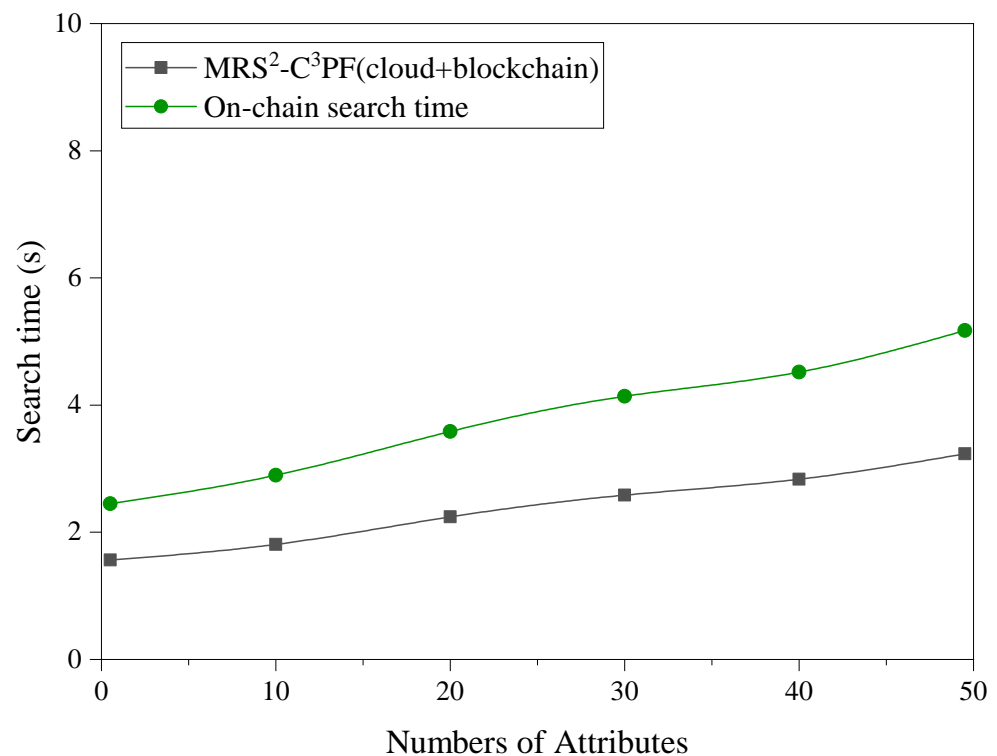


Figure 6. Search algorithm time comparison.

7.2. Experimental Analysis

The functional analysis and complexity analysis of the $MDS^2 - C^3PF$ scheme are analyzed in this section.

7.2.1. Function Comparison

Table 6 lists the functional benefits of the $MDS^2 - C^3PF$ scheme. As mentioned in the related work, the works [27–31] showed a good performance on data sharing. Reference [29] and the $MDS^2 - C^3PF$ scheme implement policy hiding. The works [27,29,31] and our scheme all realize multi-keyword search operations. However, the $MDS^2 - C^3PF$ scheme implements co-authorization to further ensure the balance of access control among data owners.

Table 6. Function comparison.

Scheme	[27]	[28]	[29]	[30]	[31]	$MDS^2 - C^3PF$
Multi-keyword search	✓	×	✓	×	✓	✓
Co-authorization	×	×	×	×	×	✓
Policy hiding	×	×	×	✓	×	✓
Data sharing	✓	✓	✓	✓	✓	✓

7.2.2. Complexity Analysis

Table 7 shows the comparison of the computational algorithm complexity in different operation stages. The works [27,28] are similar to the relevant algorithm proposed in this paper, which adopt CP-ABE and SE to solve the problem of data sharing. Therefore, they were compared with the algorithms in the $MDS^2 - C^3PF$ scheme. Compared with [27,28], the overhead of the proposed method in the setup is lower than [27]. It also has advantages over [27,28] in terms of the exponential operational overhead of the search phase. The

bilinear matching time is denoted by T_p , and T_e is the exponential time. The hash operation time is represented by T_h . Let n denote the number of attributes of the authorized user, and l is the number of attributes in the policy. m denotes the keyword count, and t denotes the number of keywords that the authorized user wants to find.

In order to make further comparisons and analysis, the runtimes of *KeyGen*, *IndexGen*, and *TrapGen* were tested through experiments. Attribute numbers were used as a variable because they can significantly affect the runtime.

The time comparison of the *KeyGen* algorithm is shown in Figure 7. From the figure, it can be seen that the key generation time rises as the number of attributes increases. Figure 8 shows the time comparison of trapdoor generation. As can be seen from the figure, the trapdoor generation time increases with the number of attributes. The key generation time and trapdoor generation time of the scheme are basically the same as those of the schemes in [27,28].

Table 7. Comparison computational overhead of our scheme.

Scheme	[27]	[28]	$MDS^2 - C^3PF$
Setup	$(3n + 1)T_e + T_p$	$3T_e$	$3T_e$
KeyGen	$(3n + 1)T_e$	$(2n + 2)T_e + nT_h$	$(2n + 3)T_e + nT_h$
IndexGen	$(3l + 3)T_e + mT_h$	$(2l + 4)T_e + (l + 1)T_h$	$T_p + (2l + 4)T_e + (m + 1)T_h$
TrapGen	$(3l + 1)T_e + T_h$	$(2n + 4)T_e + T_h$	$(t + 2)T_e + tT_h$
Search	$(3l + 1)T_p + T_e$	$(2n + 3)T_p + T_e$	$(t + 2)T_p$

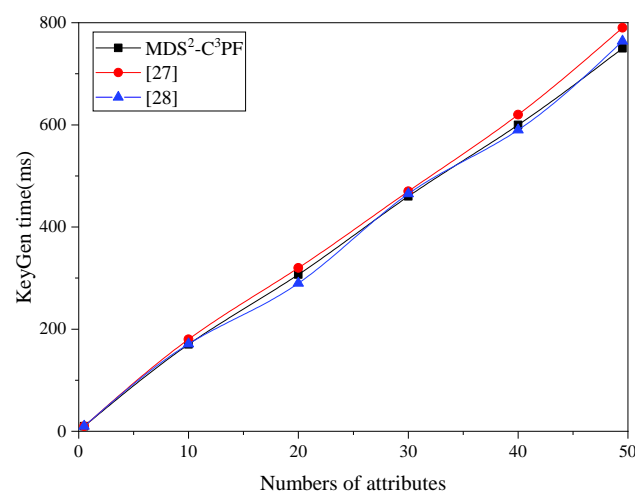


Figure 7. KeyGen algorithm time comparison.

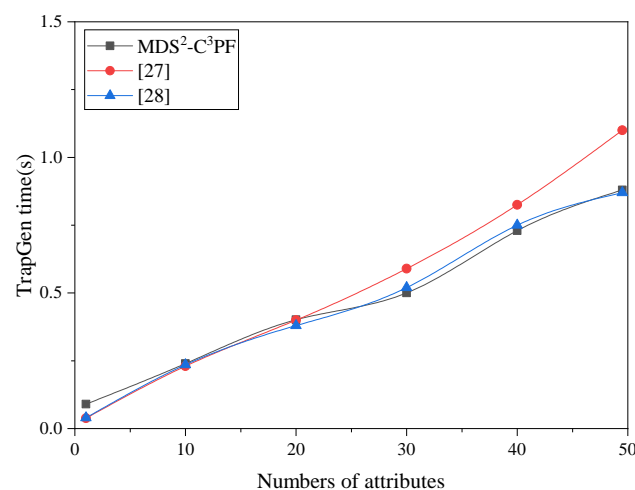


Figure 8. TrapGen algorithm time comparison.

Figure 9 shows the comparison of the index generation time. From the figure, the time of the index generation algorithm rises with increasing attribute number. Compared with [27,28], the $MDS^2 - C^3PF$ scheme implements an index generation algorithm with policy hiding to improve its security. Therefore, our *IndexGen* algorithm's time consumption is a little higher. However, the scheme in this paper can realize multiple times of searching following one generated index, which further ensures the feasibility of our solution.

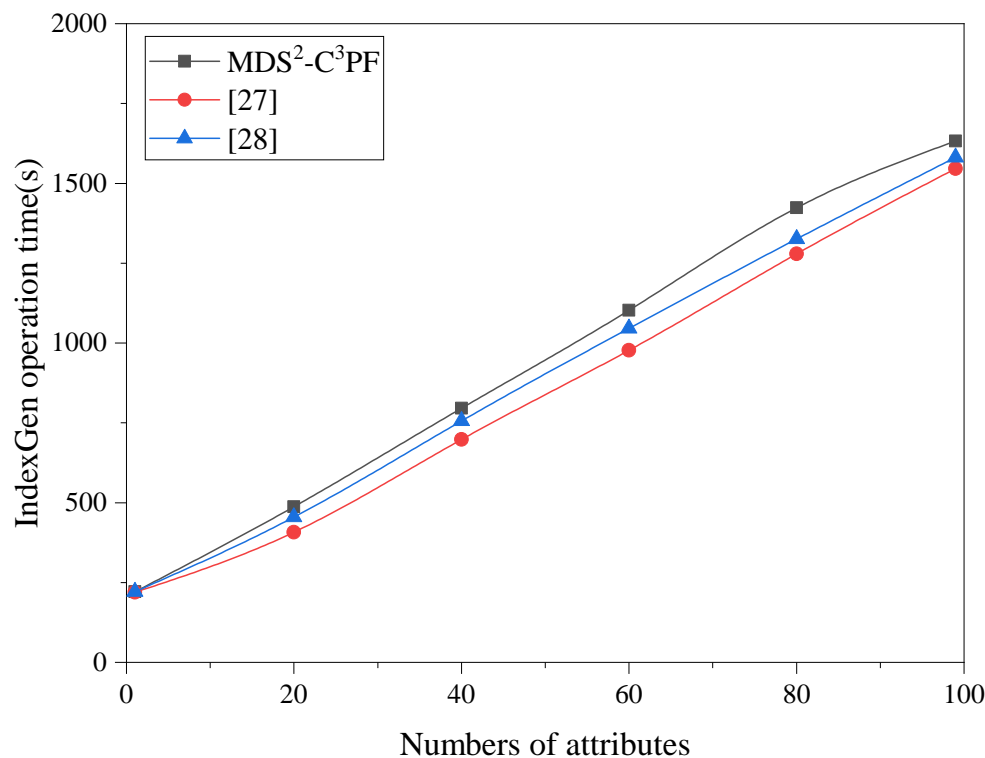


Figure 9. IndexGen algorithm time comparison.

8. Conclusions

In view of the asymmetric access control right of the medical data between doctors and patients and the asymmetric collection and processing capability of IoT terminals and the cloud, medical data sharing is faces the problems of privacy leakage, malicious tampering, and false feedback by the cloud. $MDS^2 - C^3PF$ was proposed to address these asymmetries. For data privacy, a conflict access policy fusion algorithm is used to achieve co-authorization, ensuring that doctors and patients have equal rights to control the medical data. To improve retrieval efficiency and detect spurious feedback from cloud servers, a cloud-chain cooperation retrieval scheme was proposed to balance the asymmetry structure of medical data storage and processing under IoT. Experimental results showed that our scheme can improve search efficiency and is suitable for the secure sharing of medical data with a symmetry structure. In fact, some weaknesses still exist in our work. Policy fusion and conflict resolution need to be completed by a trust center. Such a centralized approach may have some security risks. In future work, the access control policy fusion will be manipulated in a more symmetric way by using blockchain. Inter-domain dynamic access control for medical data sharing will also be further discussed.

Author Contributions: Conceptualization, H.P., Y.Z., Z.Y. and X.S.; software, Y.Z.; validation, Y.Z. and Z.Y.; formal analysis, H.P., Z.Y. and L.Z.; investigation, H.P. and Y.Z.; writing—original draft preparation, Y.Z.; writing—review and editing, H.P., Y.Z., Z.Y. and L.Z.; supervision, H.P., L.Z., Y.Z. and X.S; project administration, H.P. and X.S.; funding acquisition, H.P. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the Henan Key Laboratory of Network Cryptography Technology under Grant LNCT2022-A12 and in part by the Major Science and Technology Project of Henan Province under Grant 201300210300.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no potential conflict of interest with respect to the research, authorship, or publication of this article.

References

1. Gubbi, J.; Buyya, R.; Marusic, S.; Palaniswami, M. Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Gener. Comput. Syst.* **2013**, *29*, 1645–1660. [[CrossRef](#)]
2. Zhou, J.; Cao, Z.; Dong, X.; Vasilakos, A.V. Security and Privacy for Cloud-Based IoT: Challenges, Countermeasures, and Future Directions. *IEEE Commun. Mag.* **2017**, *55*, 26–33. [[CrossRef](#)]
3. Sahai, A.; Waters, B. *Fuzzy Identity-Based Encryption*; Springer: Berlin, Germany, 2005; pp. 457–473.
4. Bethencourt, J.; Sahai, A.; Waters, B. Ciphertext-Policy Attribute-Based Encryption. In Proceedings of the 2007 IEEE Symposium on Security and Privacy (SP '07), Oakland, CA, USA, 20–23 May 2007; pp. 321–334.
5. Yin, C.; Wang, H.; Zhou, L.; Fang, L. Ciphertext-policy attribute-based encryption with multi-keyword search over medical cloud data. In Proceedings of the 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), Guangzhou, China, 9 February 2021.
6. Hsieh, G.; Chen, R.J. Design for a secure interoperable cloud-based Personal Health Record service. In Proceedings of the 4th IEEE International Conference on Cloud Computing Technology and Science Proceedings, Taipei, Taiwan, 3–6 December 2012; pp. 472–479.
7. Sangeetha, D.; Chakkaravarthy, S.S.; Satapathy, S.C.; Vaidehi, V.; Cruz, M.V. Multi keyword searchable attribute based encryption for efficient retrieval of health Records in Cloud. *Multimed. Tools Appl.* **2022**, *81*, 22065–22085. [[CrossRef](#)]
8. Boneh, D.; Crescenzo, G.D.; Ostrovsky, R.; Persiano, G. *Public Key Encryption with Keyword Search*; Springer: Berlin, Germany, 2004; pp. 506–522.
9. Ding, Y.; Xu, H.; Wang, Y.; Yuan, F.; Liang, H. Secure Multi-Keyword Search and Access Control over Electronic Health Records in Wireless Body Area Networks. *Secur. Commun. Netw.* **2021**. [[CrossRef](#)]
10. Ramu, G.; Reddy, B.E.; Jayanthi, A.; Prasad, L.V. Fine-grained access control of EHRs in cloud using CP-ABE with user revocation. *Health Technol.* **2019**, *9*, 487–496. [[CrossRef](#)]
11. Zheng, Z.B.; Xie, S.A.; Dai, H.N.; Wang, H. Blockchain challenges and opportunities: A survey. Blockchain challenges and opportunities: A survey. *Int. J. Web Grid Serv.* **2018**, *14*, 352–375. [[CrossRef](#)]
12. Xu, C.; Fulong, C.; Dong, X.; Sun, H.; Huang, C. Design of a Secure Medical Data Sharing Scheme Based on Blockchain. *J. Med. Syst.* **2020**, *44*, 52.
13. Gai, K.; Guo, J.; Zhu, L.; Yu, S. Blockchain Meets Cloud Computing: A Survey. *IEEE Commun. Surv. Tutorials* **2020**, *22*, 2009–2030. [[CrossRef](#)]
14. Chen, L.; Lee, W.K.; Chang, C.C.; Choo, K.K.R.; Zhang, N. Blockchain based searchable encryption for electronic health record sharing. *Future Gener. Comput. Syst.* **2019**, *95*, 420–429. [[CrossRef](#)]
15. Gupta, B.B.; Li, K.C.; Leung, V.C.; Psannis, K.E.; Yamaguchi, S. Blockchain-Assisted Secure Fine-Grained Searchable Encryption for a Cloud-Based Healthcare Cyber-Physical System. *IEEE/CAA J. Autom. Sin.* **2021**, *8*, 1877–1890.
16. Han, D.; Pan, N.; Li, K. A Traceable and Revocable Ciphertext-Policy Attribute-based Encryption Scheme Based on Privacy Protection. *IEEE Trans. Dependable Secur. Comput.* **2022**, *19*, 316–327. [[CrossRef](#)]
17. Li, Q.; Xia, B.; Huang, H.; Zhang, Y.; Zhang, T. TRAC: Traceable and revocable access control scheme for mHealth in 5G-enabled IIoT. *IEEE Trans. Ind. Inform.* **2021**, *18*, 3437–3448. [[CrossRef](#)]
18. Hu, G.; Zhang, L.; Mu, Y.; Gao, X. An Expressive “Test-Decrypt-Verify” Attribute-Based Encryption Scheme with Hidden Policy for Smart Medical Cloud. *IEEE Syst. J.* **2020**, *15*, 365–376. [[CrossRef](#)]
19. Sowjanya, K.; Dasgupta, M.; Ray, S. A lightweight key management scheme for key-escrow-free ECC-based CP-ABE for IoT healthcare systems. *J. Syst. Archit.* **2021**, *117*, 102108. [[CrossRef](#)]
20. Hwang, Y.W.; Lee, I.Y. A Study on CP-ABE-Based Medical Data Sharing System with Key Abuse Prevention and Verifiable Outsourcing in the IoMT Environment. *Sensors* **2020**, *20*, 4934. [[CrossRef](#)]
21. Liu, J.; Wu, M.; Sun, R.; Du, X.; Guizani, M. BMDS: A Blockchain-based Medical Data Sharing Scheme with Attribute-Based Searchable Encryption. In Proceedings of the ICC 2021-IEEE International Conference on Communications, Montreal, QC, Canada, 14–23 June 2021; pp. 1–6.
22. Wang, S.; Wang, H.; Li, J.; Wang, H.; Chaudhry, J.; Alazab, M.; Song, H. A fast CP-ABE system for cyber-physical security and privacy in mobile healthcare network. *IEEE Trans. Ind. Appl.* **2022**, *56*, 4467–4477. [[CrossRef](#)]
23. Li, H.; Yang, Y.; Dai, Y.; Yu, S.; Xiang, Y. Achieving Secure and Efficient Dynamic Searchable Symmetric Encryption over Medical Cloud Data. *IEEE Trans. Cloud Comput.* **2020**, *8*, 484–494. [[CrossRef](#)]
24. Mingwu, Z.; Yu, C.; Jiajun, H. SE-PPFM: A Searchable Encryption Scheme Supporting Privacy-Preserving Fuzzy Multikeyword in Cloud Systems. *IEEE Syst. J.* **2021**, *15*, 2980–2988.

25. Payal, C.; Manik, L.D. Privacy Preserving Searchable Encryption with Fine-Grained Access Control. *IEEE Trans. Cloud Comput.* **2021**, *9*, 753–762.
26. Shahzaib, T.; Sushmita, R.; Yogachandran, R.; Rajarajan, M.; Glackin, C. A New Secure and Lightweight Searchable Encryption Scheme over Encrypted Cloud Data. *IEEE Trans. Emerg. Top. Comput.* **2019**, *7*, 530–544.
27. Sun, W.; Yu, S.; Lou, W.; Hou, Y.T.; Li, H. Protecting your right: Verifiable attribute-based keyword search with fine-grained owner-enforced search authorization in the cloud. *IEEE Trans. Parallel Distrib. Syst.* **2014**, *27*, 1187–1198. [[CrossRef](#)]
28. Zheng, Q.; Xu, S.; Ateniese, G. VABKS: Verifiable attribute-based keyword search over outsourced encrypted data. In Proceedings of the IEEE INFOCOM 2014-IEEE Conference on Computer Communications, Toronto, ON, Canada, 27 April 2014; pp. 522–530.
29. Liu, S.; Yu, J.; Xiao, Y.; Wan, Z.; Wang, S.; Yan, B. BC-SABE: Blockchain-Aided Searchable Attribute-Based Encryption for Cloud-IoT. *IEEE Internet Things J.* **2020**, *7*, 7851–7867. [[CrossRef](#)]
30. Liu, J.; Li, X.; Ye, L.; Zhang, H.; Du, X.; Guizani, M. BPDS: A blockchain based privacy-preserving data sharing for electronic medical records. In Proceedings of the 2018 IEEE Global Communications Conference (GLOBECOM), Abu Dhabi, United Arab Emirates, 9–13 December 2018; pp. 1–6.
31. Cao, S.; Zhang, G.; Liu, P.; Zhang, X.; Neri, F. Cloud-assisted secure eHealth systems for tamper-proofing EHR via blockchain. *Inf. Sci.* **2019**, *485*, 427–440. [[CrossRef](#)]
32. Zhang, L.; Zhang, T.; Wu, Q.; Mu, Y.; Rezaeibagha, F. Secure Decentralized Attribute-Based Sharing of Personal Health Records with Blockchain. *IEEE Internet Things J.* **2021**. [[CrossRef](#)]
33. Munagala, N.V.L.M.; Rani, A.; Reddy, D.V. Blockchain-Based Internet-of-Things for Secure Transmission of Medical Data in Rural Areas. *Comput. J.* **2022**. [[CrossRef](#)]
34. Chen, W.; Zhu, S.; Li, J.; Wu, J.; Chen, C.L.; Deng, Y.Y. Authorized Shared Electronic Medical Record System with Proxy Re-Encryption and Blockchain Technology. *Sensors* **2021**, *21*, 7765. [[CrossRef](#)] [[PubMed](#)]
35. Saini, A.; Zhu, Q.; Singh, N.; Xiang, Y.; Gao, L.; Zhang, Y. A Smart-Contract-Based Access Control Framework for Cloud Smart Healthcare System. *IEEE Internet Things J.* **2021**, *8*, 5914–5925. [[CrossRef](#)]
36. Chen, C.L.; Deng, Y.Y.; Weng, W.; Sun, H.; Zhou, M. A blockchain-based secure inter-hospital EMR sharing system. *Appl. Sci.* **2020**, *10*, 4958. [[CrossRef](#)]
37. Yang, L.; Jiguo, L. Efficient searchable public key encryption against keyword guessing attacks for cloud-based EMR systems. *Clust. Comput.* **2018**, *22*, 285–299.
38. Chen, N.; Li, J.; Zhang, Y.; Guo, Y. Efficient CP-ABE scheme with shared decryption in cloud storage. *IEEE Trans. Comput.* **2020**, *71*, 175–184. [[CrossRef](#)]
39. Xiao, Q.; Tan, K.L. Peer-aware collaborative access control in social networks. In Proceedings of the 8th International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom), Pittsburgh, PA, USA, 14–17 October 2012; pp. 30–39.