Computer Modeling in
Engineering & Sciences

Tech Science Press

DOI: 10.32604/cmes.2023.026321

ARTICLE

BC-PC-Share: Blockchain-Based Patient-Centric Data Sharing Scheme for PHRs in Cloud Computing

Caihui Lan[1] **and Haifeng Li**[2,3,*]

[1]School of Electronic and Information Engineering, Lanzhou City University, Lanzhou, 730070, China

[2]School of Software, Dalian University of Foreign Languages, Dalian, 116044, China

[3]School of Software, Dalian University of Technology, Dalian, 116024, China

*Corresponding Author: Haifeng Li. Email: lihaifengdlut@163.com

Received: 30 August 2022 Accepted: 21 November 2022

ABSTRACT

Sharing of personal health records (PHR) in cloud computing is an essential functionality in the healthcare system. However, how to securely, efficiently and flexibly share PHRs data of the patient in a multi-receiver setting has not been well addressed. For instance, since the trust domain of the cloud server is not identical to the data owner or data user, the semi-trust cloud service provider may intentionally destroy or tamper shared PHRs data of user or only transform partial ciphertext of the shared PHRs or even return wrong computation results to save its storage and computation resource, to pursue maximum economic interest or other malicious purposes. Thus, the PHRs data storing or sharing via the cloud server should be performed with consistency and integrity verification. Fortunately, the emergence of blockchain technology provides new ideas and prospects for ensuring the consistency and integrity of shared PHRs data. To this end, in this work, we leverage the consortium blockchain technology to enhance the trustworthiness of each participant and propose a blockchain-based patient-centric data sharing scheme for PHRs in cloud computing (BC-PC-Share). Different from the state-of-art schemes, our proposal can achieve the following desired properties: (1) Realizing patient-centric PHRs sharing with a public verification function, i.e., which can ensure that the returned shared data is consistent with the requested shared data and the integrity of the shared data is not compromised. (2) Supporting scalable and fine-grained access control and sharing of PHRs data with multiple domain users, such as hospitals, medical research institutes, and medical insurance companies. (3) Achieving efficient user decryption by leveraging the transformation key technique and efficient user revocation by introducing time-controlled access. The security analysis and simulation experiment demonstrate that the proposed BC-PC-Share scheme is a feasible and promising solution for PHRs data sharing via consortium blockchain.

KEYWORDS

Blockchain; patient-centric; personal health records; data sharing; attribute-based encryption

1 Introduction

A Personal Health Record (PHR) refers to an electronic record collection administratered by the patients themselves. It commonly includes healthcare information and medical information obtained from a variety of sources, such as multiple healthcare providers (e.g., hospitals) and the patients themselves (e.g., wearable devices) [1,2]. More specifically, PHR usually contains personal information, drug history (including dosages), history of drug allergy, records of previous illnesses and surgeries, chronic health problems, such as high blood pressure, family history, immunization records, lab test results, and other personal health information. A PHR system demonstrates the potential benefits of reducing misunderstandings and mistakes, avoiding unnecessary double treatments, physical examinations and medical tests, and eliminating adverse drug events, and so on in healthcare. The PHR is a patient-centric tool that can help increase patients' engagement to actively learn their own health status, easily track the course of treatment, and actively cooperate with the scientific medical treatment of the health care provider. With the assistance of PHR service, the quality of healthcare can be improved significantly. Given their enormous merits, PHR systems have attracted extensive interest worldwide. Cloud computing, as a resource provision platform, offers users mass storage space, powerful computing capability, and ubiquitous access service, which is like the traditional public utility, such as water and electricity [3]. In the pursuit of universal accessibility and low cost, in practice, the PHR service providers usually resort to the cloud server to store and share their PHRs data. Although the great potential benefits are indisputable, the adoption of PHRs data sharing via the cloud computing is hindered by data security and data privacy concerns in the healthcare domain, because the PHRs include a large amount of sensitive information of the patient. For the sake of ensuring the security of the PHRs, a large number of relevant schemes have been put forward, such as [4–8]. Nevertheless, these schemes commonly rely on a third trust authority which is difficult to find in reality, and are faced with the single point of failure issue. These security challenges urge further consideration and exploration.

Fortunately, the emerging blockchain technology can provide a promising and feasible solution for the secure sharing of PHRs data. The blockchain technology incorporates many promising characteristics such as decentralization, immutability, traceability, openness and anonymity. The intrinsic value of blockchain is to establish trust in a trustless distributed system without introducing any third trusted authority due to its unique features. In recent years, the blockchain technology has been envisioned as a significant innovation in information technology and aroused considerable interest in both the academic community and the industrial alliance. Over the past few years, sparked by the enormous achievements of blockchain in digital cryptocurrency, many scholars have endeavored to extend its application to various fields, such as Internet of Things [9–11], medical data sharing [12–15], cloud computing [16–18], and so forth.

1.1 Motivation and Contribution

Motivation. To achieve secure, efficient and flexible PHRs data sharing in multi-receiver settings, it is essential to identify the following potential attacks:

1) Unauthorized access. In a healthcare system, PHRs data requires maintenance in a secure and private environment. The patient should have complete control over his/her PHR data and only the users authorized by the patient can have the right to full or partial access to the patient's PHRs data. Any unauthorized access should be prohibited. In practice, a passive adversary or malicious cloud server may eavesdrop on the sensitive PHRs data of patient for making profits or other evil purpose. All these malicious behaviors can pose a security threat of leaking the sensitive PHR information.

2) Integrity and consistency of shared data. An active malicious adversary may try to get unauthorized right to tamper the PHRs data before an authorized user (e.g., a doctor) can access them. This vicious active attack may lead to a misdiagnosis or a wrong treatment for the patient and

cause serious harms to the patient's health. What is even worse, this vicious behavior may cause the death of the patient. Therefore, it is of the utmost importance that the integrity and consistency of the shared sensitive PHRs data of the patient is ensured.

3) Efficient revocation and decryption. To take the advantage of the premium computational ability, we introduce transformation key technique to relieve the heavy computation burden of the data users by offloading computationally intensive operations to the cloud server without leaking any sensitive data nor compromising privacy. This is especially meaningful considering the increasing popularity of resource-limited mobile devices. In addition, the efficient revocation should also be considered.

To achieve the patient-centric data sharing in a multiple receiver setting, it is essential to utilize the fine-grained access control that can support multiple domain users for accessing the PHRs data simultaneously. For instance, the patient intends to share their PHRs data to a hospital for the purpose of gaining the services of diagnosis, examination, and healthcare. And the patient can share partial non-sensitive information with medical research institute for conducting scientific research to enhance the quality of healthcare for human beings. And the patient can also share relevant information with their medical insurance company for processing of medical insurance claims.

Driven by these demands, the attribute-based encryption (ABE) scheme is introduced to guarantee the confidentiality and fine-grained access control simultaneously. However, directly using the ABE mechanism will result in several security issues. Firstly, most of the existing ABE schemes do not support the consistency and integrity checking of the shared data, and that puts the data owner and/or user at a clear disadvantage, especially in medical field, as it can lead to a misdiagnosis or other event that could endanger the patient's life. Secondly, currently, with the broad adoption of mobile devices, the utilization of the highly efficient user revocation and user decryption should be considered because most existing ABE schemes commonly involve heavy cryptographic operations.

Contribution. To fill the identified gaps, in this work, we employ the consortium blockchain technology to enhance the trustworthiness of each participant and propose a blockchain-based patient-centric data sharing scheme to achieve public verifiability, immutability, scalability and fine-grained PHRs data sharing in a multiple receiver setting. The main contributions of this work are listed as follows:

1) First, we devise a novel blockchain based system model for patient-centric secure data sharing of PHRs in cloud computing under a multiple receiver setting. In the system model, we conceptually categorize the receiver into three different domains: hospitals, medical research institutes, and medical insurance companies.

2) Second, the proposed BC-PC-Share scheme can not only ensure the integrity and consistency of the shared data via semi-trust cloud computing by adopting smart contracts to perform public consistency and integrity checking, but also achieve scalable and fine-grained access control and PHRs data sharing with multiple domain users by utilizing the ABE mechanism. Moreover, the envisaged scheme can significantly reduce the decryption cost by leveraging the transformation key technique and can achieve efficient user revocation by introducing time-controlled access (i.e., the patient indirectly revokes user's access privileges by assigning the invalid period).

3) Finally, we present a concrete security analysis of the proposed BC-PC-Share scheme in terms of the correctness, CPA security, completeness, and efficient user revocation under the random oracle model. Moreover, we conduct a simulation experiment to evaluate the performance of our proposed

BC-PC-Share solution. We also make a comparison of our blockchain based solution with several representative works and demonstrate that our scheme is both practical and efficient.

1.2 Paper Organization

The remainder of the paper is organized as follows. The related literatures are reviewed in Section 2. The background knowledge is presented in Section 3. The system framework of the BC-PC-Share scheme is modeled in Section 4. Following this, the concrete BC-PC-Share scheme is proposed in Section 5. Next, in Sections 6 and 7, security analysis and performance evaluation, are respectively conducted. Finally, the conclusion of this work is drawn in Section 8.

2 Related Work

In this section, we discuss the literature related to the PHRs sharing, including cloud-based PHRs sharing schemes and blockchain-based PHRs sharing schemes.

A. Cloud-Based PHRs Sharing Schemes

In 2013, Li et al. [4] proposed a patient-centric role-based framework for secure sharing of PHRs data in a cloud computing environment. Considering that the cloud server is not fully trusted, they adopted the ABE mechanism to achieve the confidentiality and secure sharing of PHRs data among different domain users simultaneously. Au et al. [5] suggested a general cloud-based framework for patients to fully control and securely share their sensitive PHRs data with users in the same domain and across domains. Xhafa et al. [7] proposed a multi-authority CP-ABE based PHRs data sharing scheme, which supports hidden access policy as well as the traceability and accountability of misbehaving PHR users. Despite the ABE scheme being very powerful and promising, it still suffers from a decryption efficiency weakness due to the fact that the traditional ABE based schemes involve many expensive pairing operations. To improve the decryption efficiency, many lightweight ABE data sharing schemes with outsourced decryption have been proposed [19–22]. Xiong et al. [19] proposed an A^2B^2E scheme with the features of hidden access policy, in which they utilized the verifiable outsourcing decryption technique for ABE and the technique of online/offline to reduce the computational cost. However, their scheme does not support user revocation. To enhance the computational efficiency on the user-side in a PHR system, Zhang et al. [6] devised a lightweight scheme called CCP-ABAC-UA. In their scheme, the PHR receivers can access PHRs with non-bilinear-pairing computation. Additionally, their scheme can support public auditing and user revocation.

B. Blockchain-Based PHRs Sharing Schemes

However, all the above-mentioned schemes suffer from the single point of failure issue. To facilitate sharing of PHRs data with a high level of confidence in the distribution setting, the blockchain technique is introduced as a potentially promising approach for building trustworthiness among different distributed institutions.

To improve the quality of medical diagnosis, Zhang et al. [23] utilized private blockchain and consortium blockchain technology to design a secure and privacy-preserving personal health information sharing (BSPP) system, in which by getting the trapdoors from the patient, enables the authorized doctor to have the privilege to search specific health records for certain patients. Thwin et al. [24] suggested a privacy-preserving access control model for secret PHRs data sharing, which supports granting and revoking access rights by incorporating the blockchain technology and proxy re-encryption (PRE) technique. Chen et al. [11] combined the searchable encryption technology with blockchain technology and proposed a blockchain-based searchable encryption scheme for eHealth data outsourced to the public cloud server, however, one of the critical concerning issues is that the cloud server is not fully reliable and not entirely trustworthy. Cao et al. [25] proposed a cloud-based secure eHealth scheme, named TP-EHR, to prevent the EHRs from any illegal tampering by adopting the blockchain technology, where a password-based authentication mechanism was introduced to resist password guessing attacks. Nagasubramanian et al. [26] proposed a KSIBC framework to guarantee the security and integrity of sensitive healthcare data between doctors and patients by leveraging blockchain technology. By combing the ABE and blockchain technology, Yang et al. [27] proposed an ABE keyword search scheme for blockchains, which can support users to search encrypted files over the blockchain according to their attributes. The above-mentioned blockchain-based eHealth data systems have a similar characteristic, that is, all of them do not use blockchain for storing eHealth data, while these schemes use blockchain for storing the metadata. Different from the previous schemes given above, Nagasubramanian et al. [26] and Al Omar et al. [28] store the healthcare data of patients in the blockchain directly and deploy it in the cloud computing environment.

3 Preliminaries

In this section, we present some background knowledge associated with this paper.

3.1 Blockchain

Blockchain contains continuously growing blocks, which are chronologically linked by using a set of cryptography techniques and the consensus algorithm, and thus forming a chain [29,30]. It is the underlying technology of the well-known Bitcoin which was originally developed by a mysterious person, named Satoshi Nakamoto in 2008 [31]. Essentially speaking, blockchain is a distributed publicly shared transaction ledger where transaction data is recorded permanently. As an innovative architecture, the blockchain system integrates many advanced technologies, such as a Peer-to-Peer (P2P) network, distributed ledger, and consensus mechanism as well as smart contracts. In a blockchain network, all the participating nodes are equal and collaborate with each other without the requirement of a trusted central party. Thus, it can eliminate the potential security risk of single point of failure (SPOF).

A block in blockchain is composed of two components: the block header and the block body, and the data structure of blocks is illustrated in Fig. 1. Each block in the blockchain contains a hash-pointer that points to the immediate prior block, and this basic structure produces one of the most prominent feature of the blockchain called immutability, i.e., once the data is verified by consensus nodes and stored in a blockchain, it cannot be permanently tampered with because of the irreversibility of cryptographic hash function. And the block body records the specific transaction data by using Merkle hash tree. With the time goes by, the new blocks are continuously generated and appended to the chain and once accepted by the consensus nodes, cannot be tampered with nor removed.

According to the application scenarios and the access privilege, the blockchain can be divided into three categories: public blockchain, consortium blockchain, and private blockchain.

Public blockchain refers to a blockchain in which anyone around the world can access it at any time to read data, conduct transactions, and consent nodes. Public blockchains are generally regarded as “completely decentralized” architecture because there is no individual or institution that can control or tamper with the blockchain. The public blockchain generally relies on the incentive mechanism to encourage participants to compete for bookkeeping. For instance, the well-known Bitcoin and Ethereum are two typical applications of public blockchains.

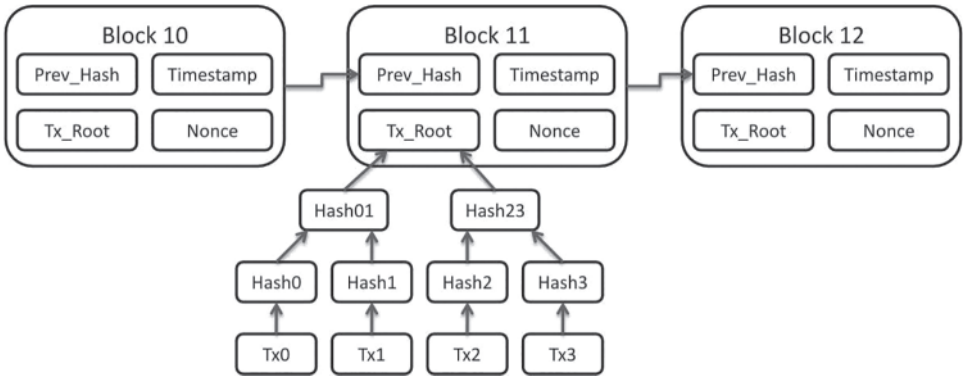

Figure 1: Data structure of blocks

Private blockchain refers to a blockchain whose write permission is controlled by an organization or institution, and the eligibility of participating nodes is severely restricted. The application scenario of private blockchain is generally the internal application of an enterprise, such as database management, and auditing.

Consortium blockchain refers to a blockchain that is managed by several institutions. Each institution runs one or more nodes. Consortium blockchain is generally considered as a "partially decentralized" platform because the data only allows participations within the system to read, write, and send transactions, and together record transaction data.

3.2 Smart Contracts

Smart contract [32] is a terminology used to describe some special script codes deployed on the blockchain that automatically executes all or parts of an agreement on all the nodes of the blockchain network among distributed untrustworthy entities without the involvement of a third trusted central party if a predefined condition is fulfilled. Smart contract is commonly an indispensably key component of the applications based on blockchain or distributed ledger technology. When predetermined conditions are met, the smart contracts automatically trigger the execution of an agreement. Moreover, they can also trigger the next action to support automatic continuous execution, namely, to form a workflow. To prevent contract from being tampered with, the smart contracts are copied to each node of the blockchain network. To further prevent contracts from being tampered with, each node also holds a copy of the smart contracts and executes the codes. The execution result of the smart contract is visible to each participant node and is verified by all participants. Only when all the participants agree on the result will they update their ledger. By this mechanism, it can yield a correct, immutable, and credible result, and further can avoid any disputes regarding the execution result of smart contracts. Thus, the smart contract can be regarded as a public trusted mechanism for correctness, but not for privacy [11].

3.3 Access Structure

Definition 3.1 (Access Structure). Let $\{P_1, P_2, \cdots, P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1,P_2,\cdots,P_n\}}$ is monotone if $\forall B, C$: if $B \in \mathbb{A}$ and $B \subseteq C$ then $C \in \mathbb{A}$. An access structure (respectively, monotone access structure) is a collection (respectively, monotone collection) $\mathbb{A}$ of non-empty subsets of $\{P_1, P_2, \cdots, P_n\}$, i.e., $\mathbb{A} \subseteq 2^{\{P_1,P_2,\cdots,P_n\}} \backslash \{\emptyset\}$. The sets in $\mathbb{A}$ are called the authorized sets. Otherwise, the sets are called the unauthorized sets.

It should be noted that, in our work, the role of the parties is represented by the attributes. Therefore, the access structure $\mathbb{A}$ will consist of the authorized sets of attributes. From now on, unless otherwise indicated, an access structure is referred to a monotone one in our context.

3.4 *Access Tree*

An access tree is used to describe an access structure [33]. In this subsection, a brief description of an access policy tree is given below:

T: This represents an access tree representing the access structure.

x: This represents a node in the access tree T, which can be categorized into two types: Leaf node and non-leaf node (interior node). Each non-leaf interior node is represented by a threshold gate, such as "AND" or "OR" threshold gate while each leaf node is associated with an attribute.

num_x: This represents the number of children of the node x.

k_x: This represents the threshold value of node x, where $0 \leq k_x \leq num_x$. If $k_x = 1$ and x is an interior node, it means that the threshold is an "OR" gate. If $k_x = num_x$ and x is an interior node, it means that the threshold is an "AND" gate. In particular, the threshold value of each leaf node x is defined as $k_x = 1$.

$parent(x)$: The function $parent(x)$ is used to return the parent of the node x in the access tree.

$index(x)$: The function $index(x)$ is used to return a unique number associated with the node x, where the number is uniquely assigned to x in a certain manner.

$att(x)$: The function $att(x)$ is used to return an attribute associated with the leaf node x in the access tree.

T_x: This represents the sub-tree for T rooted at the node x in the access tree.

If an attribute set S matches the sub-access-tree T_x, it is represented as $T_x(S) = 1$. $T_x(S)$ outputs 1 if and only if the following conditions are satisfied:

(1) If x is a leaf node, $T_x(S) = 1$ if and only if $att(x) \in S$.

(2) If x is an interior node, each child $T_z(S)$ of node x is individually computed in a recursive way. $T_x(S) = 1$ if and only if at least k_x children return 1.

3.5 *Attribute-Based Encryption*

Attribute-based encryption(ABE) is a hot direction in cryptographic research in recent years. It can effectively realize fine-grained non-interactive access control mechanisms and has a broad range of application prospects. In 2005, Sahai et al. [34] first developed a new idea of a fuzzy identity-based encryption scheme (Fuzzy-IBE), in which the unique identity information was extended to biological characteristic information. And later, it was developed to the notion of attribute-based encryption by Goyal et al. [35]. On the base of the combination approach of the secret key and ciphertext associated with the access policy, the ABE schemes can be mainly classed as Key-Policy ABE(KP-ABE) scheme [35] and Ciphertext-Policy ABE(CP-ABE) scheme [33]. In the KP-ABE mechanism, the secret keys are bound with an access policy and the ciphertext are bound with a set of attributes. In contrast, in the CP-ABE mechanism, the secret keys are associated with a set of attributes and the ciphertext are bound with an access policy. In the CP-ABE scheme, the data owners can define access policies over ciphertexts arbitrarily, therefore, it is more suited for fine grained access control than KP-ABE scheme in cloud computing. By virtue of this merit, a number of CP-ABE schemes for securely sharing medical data have been proposed in cloud computing [4,36–38].

3.6 Bilinear Pairing

Definition 3.2 (Bilinear Pairing [22]). Let G_1, G_2 and G_T be three multiplicative cyclic groups with the equal prime order p and assume g_1 and g_2 are generators of G_1, and G_2, respectively. A map e: $G_1 \times G_2 \to G_T$ is defined as a bilinear map if and only if the following three properties hold:

(1) Bilinearity. For $\forall g_1 \in G_1$, $g_2 \in G_2$ such that $e({g_1}^a, {g_2}^b) = e(g_1, g_2)^{ab}$, where a, $b \in Z_p$ are two random numbers and Z_p is a finite field.

(2) Non-degeneracy. $\exists g_1 \in G_1$, $g_2 \in G_2$ such that $e(g_1, g_2) \neq 1_{G_T}$, where 1 is the identity element of group G_T.

(3) Computability. For $\forall g_1 \in G_1$, $g_2 \in G_2$, there exists an efficient algorithm to compute $e(g_1, g_2)$.

3.7 Complexity Assumptions

Definition 3.3 (Decision Bilinear Diffie-Hellman Problem (BDDHP) [22]). Let G and G_T be two multiplicative cyclic groups with the equal prime order p, e: $G \times G \to G_T$ is a bilinear map, and Z_q^* is a finite field. Given a quadruple $(g, g^a, g^b, g^c) \in G$ and $e(g, g)^d \in G_T$, where g is a generator of G, a, b, $c \in_R Z_q^*$ are unknown, $d \in_R Z_q^*$, determine whether $e(g, g)^d = e(g, g)^{abc} \in G_T$ or $d \equiv abc \in_R Z_q^*$ holds.

4 System Model

In this section, we discuss the system model of the proposed BC-PC-Share scheme, including the system architecture, the workflow of the proposed scheme and the general definition of the BC-PC-Share scheme.

4.1 System Architecture

In this work, we consider a specific scenario that a patient wants to share his/her PHRs data to multiple organizations, such as hospitals, medical research institutes, and medical insurance companies for different uses and purposes. Thus, we adopt a consortium blockchain to reserve the authentication information and achieve the patient-centric secure medical data sharing. Accordingly, as illustrated in Fig. 2, the system model of the proposed BC-PC-Share scheme mainly consists of four entities: the Data Owner (DO), the Data User (DU), the Cloud Service Provider (CSP), and the BlockChain (BC). The patient-centric data sharing framework can be divided into three different layers, named Data Layer, User Layer and Authentication layer, respectively. The function of each layer is discussed as follows:

1) Data Layer. The function of this layer is to provide the storage service for the data owner DO and provide the downloading service for the data user DU. There is only one entity in the Data Layer.

• CSP. The CSP is responsible for storing the patient's encrypted PHRs data uploaded by the patient, and it is also responsible for simplifying the complicated ciphertext and providing the service of downloading the shared data for the authorized data user DU. Note that the CSP is generally regarded as a semi-trusted third party because its trust domain is not identical to the DO or DU. Especially, the cloud computing provider may intentionally destroy or tamper with the user's sensitive data or only transform partial ciphertext of the shared PHRs or even return wrong computation results in a bid to save its storage and computation resource to pursue maximum economic interest, or other malicious purpose. Thus, the data storing to the CSP or the shared data downloading from the CSP should be performed with integrity and consistency verification and the validity of the ciphertext transformed by the CSP should also be checked.

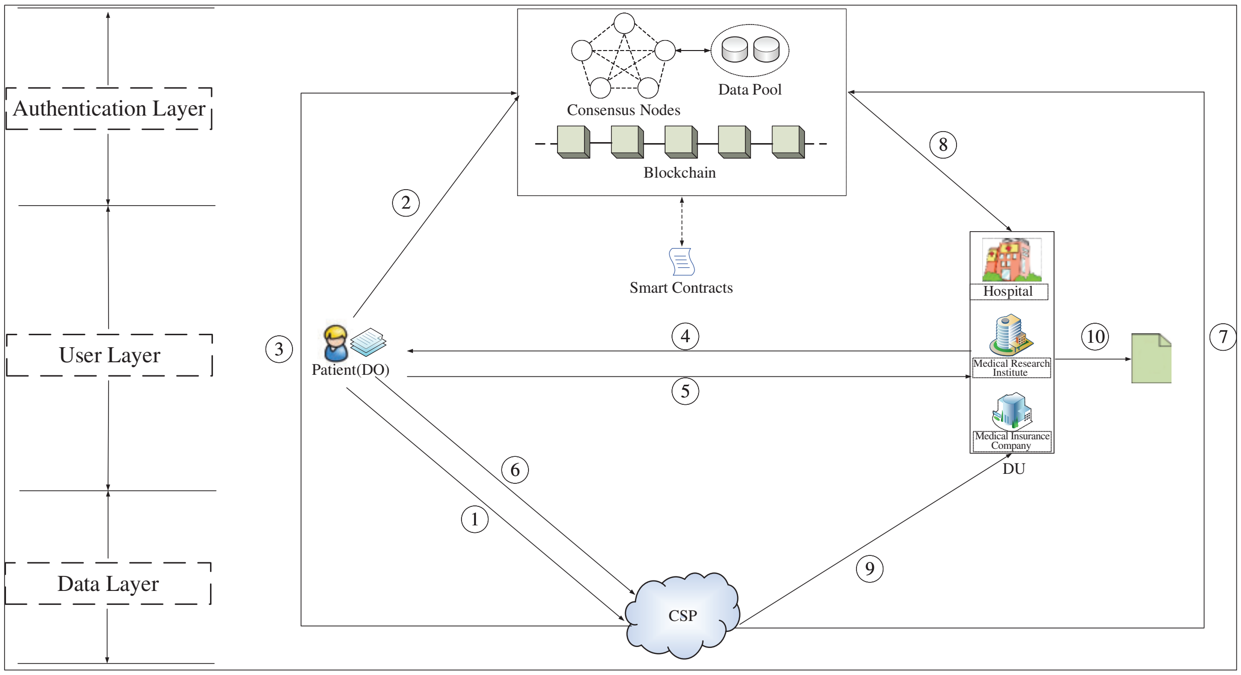

Figure 2: System model of BC-PC-Share

2) User Layer. The components of this layer include two entities: the data owner DO and the data user DU. Their roles in our scheme can be described as below:

• DO. In the proposed BC-PC-Share scheme, the DO is referred to as the patient owning a series of PHR data and with full control over his/her PHR data for sharing them with hospitals, medical research institutes, medical insurance companies and other entities.

• DU. The DU is referred to the individuals or organizations that are allowed to access the patient's PHR data for medically relevant purposes. Only the authenticated users can access the specified PHR data of the patient. In the proposed BC-PC-Share scheme, the DU is categorized into three different types according to their different uses of the PHR data: the hospital (for the purpose of diagnosis, examination, and healthcare for the patient himself/herself), the medical research institute (for the purpose of conducting scientific research to enhance the quality of healthcare for human beings), and the medical insurance company (for the purpose of providing medical insurance services).

3) Authentication layer. The component of this layer only includes one entity, named the Blockchain. The function of this layer is to record the related authentication information during the PHR data storage and sharing process by the blockchain network. The blockchain plays an essential role in our scheme. It is a peer-to-peer network composed of consensus nodes, a data pool, miners and so on. The consortium blockchain is introduced in our system to keep the authentication information and achieve secure data storage and data sharing.

4.2 *Workflow of the BC-PC-Share Scheme*

The interaction between each entity is shown in Fig. 2, and the specific workflow of the BC-PC-Share scheme (the description of each numbered step) in Fig. 2 can be described as follows. ① The patient (PHR data owner, DO) encrypts his/her PHR data to be outsourced and uploads the generated ciphertext to the cloud server for storage.

② The DO generates the storage authentication information for the outsourced PHR data and sends it to the smart contracts for storage verification.

③ The cloud server generates the storage validation request and sends it to the smart contracts for storage validation verification. If the storage verification is passed, the smart contracts send the storage authentication information as a transaction to the blockchain.

④ The DU sends a data sharing request with his/her attributes to the DO.

⑤ After successful verification of the validity of the request of DU, the data owner DO generates a user decryption key for DU and sends it to the DU.

⑥ At the same time, the DO generates a cloud-assisted transformation key for the DU and sends it to the cloud server.

⑦ After transforming the complicated ciphertext to a simplified ciphertext, the cloud server sends the transformation authentication information to the smart contracts for transform validation verification. If the transform verification is passed, the smart contracts send the storage authentication information as a transaction to the blockchain.

⑧ The consensus nodes in the blockchain inform the data user DU to download the shared PHR data from the cloud server.

⑨ The DU downloads the encrypted sharing data from the cloud server.

⑩ The DU decrypts the untampered encrypted sharing data and obtains the correct plain message.

4.2.1 Formal Definition

Definition 4.1 (BC-PC-Share). The proposed scheme involves the following two stages: PHR data storage stage and PHR data sharing stage.

• PHR data storage stage

The PHR data storage stage contains six procedures, i.e., Signingkey Generation, Ciphertext Generation, Storage Authentication Generation, Storage Validation Request, Storage Verification, Block Generation, and Storage Confirm, respectively. The brief description of each procedure is given below:

1) Signingkey Generation.

The DO randomly generates a signing key pair.

2) Ciphertext Generation.

The DO defines an access structure (policy), and encrypts the plaintext message to generate the corresponding ciphertext.

3) Storage Authentication Generation.

The DO sends the ciphertext to the cloud server for storage and sends the corresponding storage authentication information to the smart contracts, respectively.

4) Storage Validation Request.

The cloud server generates the storage validation request and sends it to the smart contracts for storage validation

5) Storage Verification.

The smart contracts automatically execute the storage verification. If the storage verification is passed, the smart contracts send the storage authentication information as a transaction to the blockchain.

6) Block Generation.

The consensus nodes in the blockchain run the consensus algorithm to verify the transaction which is sent by smart contracts and generate a valid block linked to the blockchain, then return the block ID to the data owner DO and the cloud server, respectively.

7) Storage Confirm.

According to the information of the returned block ID, the DO can confirm whether his/her data has been stored in integrity in the cloud server.

• PHR data sharing stage

The PHR data sharing stage involves six procedures, named, Signingkey Generation, Sharing Request Generation, Sharing Request Authorization, PHRs Transformation and Sharing, Transformation Verification and Block Generation, and Decryption, respectively. The brief description of each procedure is given below:

1) Signingkey Generation.

The data user DU generates a random signing key pair.

2) Sharing Request Authorization.

The DU sends a data sharing request to the DO.

3) Sharing Request Authorization.

The DO generates the cloud-assisted transformation key TK and the user decryption key UK, then sends the former to the cloud server and sends the latter to the DU, respectively.

4) PHR Transformation and Sharing.

The cloud server calls the $Transform_{out}$ algorithm to transform the complicated ciphertext to a simplified ciphertext. Next, the cloud server sends the transformation authentication information to the smart contracts for transform validation verification.

5) Transformation Verification and Block Generation.

The smart contracts verify the integrity and consistency of the shared PHR information (i.e., the validity of the transformed ciphertext given by the cloud server). If the transformation verification is passed, the smart contracts send the transformation authentication information as a transaction to the blockchain. Then, the consensus nodes in the blockchain perform the consensus algorithm (e.g., PBFT) to verify the transaction and generate a valid block linked to the blockchain.

6) Decryption.

After the above successful verification, the DU can decrypt the untampered encrypted PHR data with the user decryption key obtained from the DO and recover the plain message.

5 The Concrete BC-PC-Share Scheme

5.1 Scheme Description

In this section, we first design a novel cloud-assisted decryption algorithm of CP-ABE with public third-party verification in subsection A, then we propose the concrete blockchain-based patient-centric data sharing scheme for PHRs in cloud computing in subsection B.

A. Attribute-Based Encryption with Public Verifiable Outsourced Decryption (PVOD-ABE)

Lai et al. [39] proposed a novel model of CP-ABE with verifiable outsourced decryption to guarantee the correctness of the outsourced ciphertext transformation by an untrusted third party (e.g., the remote cloud server). Undoubtedly, the feature of verifiability is a tremendous progress for outsourcing ABE schemes. Nevertheless, their scheme only supported the data user to check the validity of the outsourced ciphertext transformation and failed to support the public verification. Thus, their scheme is not appropriate for the blockchain scenarios. To fill this gap, in this subsection, we develop an Attribute-Based Encryption with Public Verifiable Outsourced Decryption (PVOD-ABE) algorithm, which cannot only achieve the goal of outsourcing the majority of the intensive decryption operations to the cloud server to reduce the computational burden of the classical CP-ABE scheme, but also support the good security property of public verification.

Now, we present the details of our proposed PVOD-ABE algorithm, which consists of five steps, that is, *SystemSetup*, *KeyGen*, *Encrypt*, $Transform_{out}$, and *UserDecrypt*. Each of them is described as follows:

- $SystemSetup(1^k, U) \rightarrow (MSK, PP)$. This step is performed by the DO. Given a system security parameter k and the universal attribute set U, it first chooses two multiplicative cyclic groups G_1, G_2 with the same prime order q ($\geq 2^k$), a bilinear map e: $G_1 \times G_1 \rightarrow G_2$ between the two group G_1, G_2, and three secure one-way hash functions H_0: $Z_q^* \rightarrow G_1$, H_1: $G_2 \rightarrow \{0,1\}^k$ and H_2: $\{0,1\}^* \rightarrow \{0,1\}^k$. Then, it defines the Lagrange coefficients as $L_{i,K}(x) = \prod_{j \in K, j \neq i} \frac{x-j}{i-j}$, where $i \in Z_q^*$, and the K is a set with the elements in Z_q^*. Next, it randomly picks $g, g_2 \in G_1$ and $\alpha, \beta \in Z_q^*$, then computes $g_1 = g^{\alpha}$, $g_3 = g_2^{\alpha^{-1}\beta}$. Denote the master public key and the master secrete key pair as ($MPK = \{g, g_1, g_2, g_3\}$, $MSK = \{\alpha, \beta\}$). Finally, the DO publishes the system public parameters $PP = \{G_1, G_2, H_0, H_1, H_2, g, g_1, g_2, g_3\}$, and keeps the master secrete key MSK confidentially.
- $KeyGen(MSK, PP, S) \rightarrow (TK, UK)$. Upon receiving the master secrete key $MSK = \{a, \beta\}$, the system public parameters PP, and the attribute set of user S, the key generation algorithm firstly selects three random numbers $t, k, d \in Z_q^*$, then for each attribute, randomly chooses a number $u_i \in Z_q^*$ and computes $TK_i = (TK_{i,1}, TK_{i,2})$, where $TK_{i,1} = g_2^{t(\alpha+d)}(g_1^i H_0(\omega_i))^{tu_i}$, $TK_{i,2} = g^{tu_i}$. Finally, it returns the transform key $TK = (\{TK_i\}_{i \in S_U}, g_1^{t\beta^{-1}d}, g^t, \alpha^{-1}\beta k, g^{tk})$ and the user decryption key $UK = t$.
- $Encrypt(PP, T, m) \rightarrow (CT)$. On input the system public parameters PP, the access policy tree T, and the message $m \in \{0,1\}^k$ to be encrypted, the data owner DO generates the ciphertext CT as follows:

(1) The DO randomly selects $r \in Z_q^*$, and assigns a polynomial q_x with the degree $d_x = k_x - 1$ in a top-down approach, and sets $q_{root}(0) = r$, where the node *root* is the root of the access policy tree T. Otherwise, for non-root node x, the DO sets $q_x(0) = q_{p(x)}(index(x))$, where $p(x)$ denotes the parent node of x, and $index(x)$ denotes a unique number assigned to x in an arbitrary order.

(2) Calculate $C_x = (C_{x,1}, C_{x,2})$ for each leaf node x, where $C_{x,1} = g^{qx(0)}$, $C_{x,2} = (g_1^{att(x)} h_x)^{qx(0)}$ and $h_x = H_0(att(x))$. Here, the function $att(x)$ is used to return an attribute associated with the leaf node x in the access tree T.

(3) The DO computes $K = H_1(e(g_1, g_2)^r)$, $C_0 = m \oplus K$.

(4) Finally, the DO outputs $CT = (T, C_0, C_1, C_j^*)$.

- $Transform_{out}(PP, CT, TK) \rightarrow (CT')$

This algorithm is used to transform the complicated ciphertext CT to a simplified ciphertext CT' by the cloud server. The specific steps for the transform process are as follows:

(1) The cloud server defines a recursive algorithm $Dec(CT, TK, x)$. It takes the ciphertext CT, the cloud-assisted transformation key TK, and a node x in the access tree T as input.

(a) If x is the leaf node, then

(1) If $\omega_j = att(x) \in S$, then return as formula (1).

$$\begin{aligned} Dec(CT, TK, x) &= \frac{e(g^{qx(0)}, g_2^{t(\alpha+d)}(g_1{}^{\omega j} H_0(\omega_j))^{tu_j})}{e((g_1{}^{\omega j} H_0(\omega_j))^{qx(0)}, g^{tu_j})} \\ &= e(g_1, g_2)^{tqx(0)} e(g, g_2^d)^{tqx(0)} \end{aligned} \tag{1}$$

(2) If $att(x) \notin S$, then return $Dec(CT, TK, x) = \perp$.

(b) For node x is an internal node of access tree T, for all the children z of x, if the number of nodes which satisfy $Dec(CT, TK, z) \neq \perp$ is less than the threshold k_x, then return $Dec(CT, TK, x) = \perp$. Otherwise, randomly chooses k_x child nodes such that $Dec(CT, TK, z) \neq \perp$ to construct a node set $S'_x = \{i = index(z) | z \in S'_x$, then continue to calculate as formula (2).

$$\begin{aligned} Dec(CT, TK, x) &= \prod_{z \in S'_x} Dec(CT, TK, z)^{L_{i,S_x}(0)} \\ &= e(g_1, g_2)^{tqx(0)} e(g, g_2{}^d)^{tqx(0)} \end{aligned} \tag{2}$$

(2) The cloud server obtains $temp' = e(g_1, g_2)^{tr} e(g, g_2{}^d)^{tr}$ by calling $Dec(CT, TK, root)$, where *root* is the root node of the access tree T. Further, we could get $temp = \dfrac{temp'}{e(g_1^{t\beta^{-1}d}, g_3^r)} = e(g_1, g_2)^{tr}$.

(3) The cloud server returns the simplified ciphertext $CT' = (temp, C_0, C_1)$.

- $UserDecrypt(PP, UK, CT') \rightarrow (m)$

On input the public parameters PP, and the transformed cyphertext CT', the DU first verify the validity of the outsourced decryption by checking whether $temp^{\alpha^{-1}\beta k} = e(C_1, g_1^{tk})$ holds or not. If the equation holds, it means that the transformed ciphertext is valid. After the verification is passed, the DU computes $K = H_1(temp^{t^{-1}})$, then the DU decrypts the ciphertext and obtains the plaintext message $m = C_0 \oplus K$. Actually, the verification process can be performed by a public verifier because the verification equation is independent with the decryption key.

Correctness Proof:

According to formulas (1) and (2), the value of *temp* in the transformed ciphertext CT' can be computed as formula (3).

$$\begin{aligned} temp &= \frac{Dec(CT, TK, root)}{e(g_1^{t\beta^{-1}d}, g_3^r)} \\ &= e(g_1, g_2)^{tr}. \end{aligned} \tag{3}$$

Then, the data user DU can gain the plain message m by the user decryption key UK as follows. $m = C_0 \oplus H_1(e(g_1, g_2)^r) = C_0 \oplus H_1(temp^{t^{-1}})$. Therefore, the data user DU can recover the shared data. Hence, the correctness of the PVOD-ABE is proved.

B. The Construction of the BC-PC-Share Scheme

On the base of the PVOD-ABE algorithm in subsection A, we propose a blockchain-based patient-centric data sharing scheme for PHR data in cloud computing, which consists of two procedures: PHR data storage stage and PHR data sharing stage. To fulfill the automatic public verification, we resort to smart contracts in the proposed BC-PC-Share scheme. In addition, only the patient (i.e., the owner of the smart contract) can deploy the smart contracts on the blockchain, where the entire content of a contract is transparent to all participating nodes in the blockchain and each node can interact with the smart contract. It should be noted that all involved transactions below will automatically trigger the execution of the smart contracts if predefined conditions are satisfied. And the execution result of smart contracts is correct, immutable, and credible to all nodes as long as the security of the blockchain is guaranteed. In this sense, it cannot only achieve public verification, but also reduce the burden of data integrity and consistency checking for the patient and multiple data users.

• PHR data storage stage

In the PHR data storage stage, we introduce a hash linked list to record the storage authentication information of each file in the blockchain and use "link" as a pointer to the head of the hash list for supporting scalable storage verification by utilizing the smart contract. Specifically, the patient (DO) can outsource their PHR data to the cloud server by performing the following seven steps, namely Signingkey Generation, Ciphertext Generation, Storage Authentication Generation, Storage Validation Request, Storage Verification, Block Generation, and Storage Confirm, respectively. Each of them is described in detail as follows:

(1) Signingkey Generation.

The data owner DO first runs *SystemSetup* algorithm of the PVOD-ABE to generate the system public parameters PP, and generates a random signing key pair (ssk_O, spk_O).

(2) Ciphertext Generation.

Given a plaintext file m, the DO defines an access policy tree T, then generates the corresponding ciphertext CT by calling the *Encrypt* algorithm of the PVOD-ABE.

(3) Storage Authentication Generation.

The data owner DO calculates $R = H_2(CT)$, $\sigma_O = SSig_{ssk_O}(link, R, time, spk_O, pk_C)$, and let $link = H_3(prelink, H_2(CT), time, ID)$, where the pointer *prelink* is pointed to the head of the previous linked hash list, which is introduced for facilitating retrieval of the storage authentication information of the entire user's data stored in the cloud server at different times by connecting them as a complete linked hash list. The pointer *link* is initially a fixed-length string (e.g., all "1" strings). pk_C is the public key of the cloud server. *time* is a timestamp representing the signing time. *ID* is the block label for storing $H_2(CT), T, prelink, time$. The data owner DO keeps the value of *link* locally. At last, the DO sends the ciphertext CT to the cloud server for storage and the corresponding storage authentication information $Authen1 = (\sigma_O, link, R, spk_O, pk_C, time)$ to the smart contracts, respectively.

(4) Storage Validation Request.

Upon receiving the ciphertext CT, the cloud server generates the storage validation request $\sigma_C = (SSig_{ssk_C}(H_2(CT)), H_2(CT))$, and finally sends it to the smart contracts for storage validation verification.

(5) Storage Verification.

The storage validation request from the cloud server will automatically trigger the deployed smart contracts to execute the storage validation by determining whether $H_2(CT) = R$ holds or not. If it holds, the smart contracts confirm that the relevant PHRs information has been stored intact on the cloud server intactly, i.e., the storage verification is passed. Finally, the smart contracts send the storage authentication information $Authen1$ as a transaction to the blockchain.

(6) Block Generation.

The consensus nodes in the blockchain perform the consensus algorithm (e.g., PBFT) to verify the transaction and generate a valid block linked to the blockchain and return the block ID to the data owner DO and the cloud server, respectively.

(7) Storage Confirm.

According to the information of the returned block ID, the data owner DO can confirm his/her data has been stored in integrity in the cloud server.

• PHR data sharing stage

In the PHR data sharing stage, we leverage the blockchain to achieve a publicly verifiable data sharing scheme between the DO and the DU, i.e., the DU can verify the integrity and consistency of the shared PHR data. Specifically, the PHR data sharing stage is composed of six procedures, namely, Signingkey Generation, Sharing Request Generation, Sharing Request Authorization, PHR Transformation and Sharing, Transformation Verification and Block Generation, and Decryption, respectively. Each of them is described in detail as follows:

(1) Signingkey Generation.

The data user DU generates a random signing key pair (ssk_U, spk_U).

(2) Sharing Request Generation.

The data user DU calculates $\sigma_U = SSig_{ssk_U}(S_U, period)$ as a data sharing request and sends it to the data owner DO.

(3) Sharing Request Authorization.

After successful verification of σ_U, the DO firstly calculates $\sigma'_O = SSig_{ssk_O}(\sigma_U||link||pk_C||\alpha^{-1}\beta k, g_1^{tk})$, and calls the *KeyGen* algorithm of the PVOD-ABE to generate the cloud-assisted transformation key TK and the user decryption key UK, then sends $(\sigma'_O||\sigma_U||TK)$ to the cloud server and $(\sigma'_O||\sigma_U||UK)$ to the data user DU, respectively.

(4) PHR Transformation and Sharing.

Upon receiving the message $(\sigma'_O||\sigma_U||TK)$ from the DO, the cloud server verifies its valid and whether *period* is within the validity period, the cloud server selects the corresponding ciphertext $C = (CT_1, CT_2, \cdots, CT_n)$ which satisfy the attribute set S_U of DU, and calls the $Transform_{out}$ algorithm of the PVOD-ABE to transform the complicated ciphertext $C = (CT_1, CT_2, \cdots, CT_n)$ to a simplified ciphertext $C' = (CT_1', CT_2', \cdots, CT_n')$. Thus, migrating the majority of the heavy computation costs of the DU to the cloud server and thus greatly reducing the decryption overhead of the DU. Next, the cloud server sends the transformation authentication information $Authen2 = SSig_{ssk_C}(\sigma'_O||\sigma_U||link||\alpha^{-1}\beta k||AD(C)||AD(C'))$ to the smart contracts for transform validation verification.

(5) PHRs Verification and Block Generation.

Since the cloud server is a semi-trusted third party and it may return tampered data to the data user, it is therefore essential to verify the integrity and consistency of the shared PHR information, i.e., the returned shared data should be consistent with the requested shared data and the integrity of the shared data is not tampered. Concretely, it will automatically trigger the deployed smart contracts to perform integrity and consistency verification of $C = (CT_1, CT_2, \cdots, CT_n)$ by comparing the transformation authenticate information $Authen2$ with the hash value retrieved from the hash list pointed by the pointer *link* in blockchain. If the hash values retrieved from the blockchain is equal to the hash values sent by the cloud server in $Authen2$, it means that the validity of the transformed ciphertext given by the cloud server can be guaranteed. Then, the consensus nodes in the blockchain perform the consensus algorithm (e.g., PBFT) to verify the transaction and generate a valid block linked to the blockchain.

Subsequently, the smart contracts verify the validity of the outsourced decryption by checking whether $temp^{\alpha^{-1}\beta k} = e(C_1, g_1^{tk})$ holds or not. If the equation holds, it means that the transformed ciphertext is valid. Then, the consensus nodes in the blockchain perform the consensus algorithm (e.g., PBFT) to verify the transaction and generate a valid block linked to the blockchain.

(6) Decryption.

After the above successful verification, the data user DU downloads the encrypted sharing data from the cloud server and decrypts the untampered encrypted PHRs data with the user decryption key UK obtained from the DO and get the plain PHRs data as follows: $K = H_1\left(temp^{t^{-1}}\right), m = C_0 \oplus K$. Otherwise, outputs $\perp$.

6 Security Analysis

6.1 Security Model

Prior to proving the security of the proposed scheme, we first describe the security model for our work, which is adapted from the security model of Lai et al. [39]. To formalize the security model, the adaptive Chosen Plaintext Attacks (CPA) security game between a challenger $\mathscr{C}$ and an adversary $\mathscr{A}$ is defined as follows:

- **Setup.** The challenger $\mathscr{C}$ runs *SystemSetup* algorithm with the security parameter k and attribute universe U to generate the system public parameters PP, and returns them to the adversary $\mathscr{A}$.
- **Query Phase I.** The attacker $\mathscr{A}$ adaptively issues the following queries:

– *TK* query. Given an attribute set S from $\mathscr{A}$, the challenger $\mathscr{C}$ runs *KeyGen* algorithm to generate the transformation key *TK* and returns it to the adversary $\mathscr{A}$.

– *UK* query. Given an attribute set S from $\mathscr{A}$, the challenger $\mathscr{C}$ runs *KeyGen* algorithm to generate the user decryption key *UK* and returns it to the adversary $\mathscr{A}$.

– $Transform_{out}$ query. Given a ciphertext CT, the challenger $\mathscr{C}$ runs $Transform_{out}$ algorithm to generate the transformed ciphertext CT' and returns it to $\mathscr{A}$.

- **Challenge.** Once **Query Phase I** is over, the attacker $\mathscr{A}$ submits two messages m_0, m_1 with equal length and one access structure T^*, the challenger $\mathscr{C}$ randomly selects $b \in \{0, 1\}$, then returns the challenge ciphertext CT^* to the adversary $\mathscr{A}$.
- **Query Phase II.** The adversary $\mathscr{A}$ continues to adaptively issue the queries in **Query Phase I** with the following two restrictions:

1) The adversary $\mathscr{A}$ cannot simultaneously make the *TK* Query and the *UK* Query on the attribute sets that satisfy the challenging access structures T^*.
2) If the adversary $\mathscr{A}$ has made the *TK* Query, he/she cannot make the *UK* Query on the attribute sets that satisfy the challenging access structures T^* and vice versa.

- **Guess.** At the end of the game, the adversary $\mathscr{A}$ outputs a guess $b' \in \{0, 1\}$ for b and the adversary $\mathscr{A}$ succeeds in the security game if and only if $b' = b$.

6.2 *Security Proof*

In this subsection, we discuss the security of the proposed BC-PC-Share scheme under the general assumptions and the random oracle model, including CPA security, completeness, and efficient user revocation.

Theorem 6.1. Provided that the hardness assumption of the Decisional Bilinear Diffie-Hellman Problem (DBDHP) is intractable, the proposed BC-PC-Share scheme can achieve CPA security.

Proof. Since our devised BC-PC-Share scheme is constructed on the basis of the aforementioned PVOD-ABE algorithm, the data confidentiality of the BC-PC-Share scheme is guaranteed by the PVOD-ABE scheme. The CPA security proof of the PVOD-ABE scheme is described as a security game between a challenger $\mathscr{C}$ and an adversary $\mathscr{A}$. Given an instance of the DBDH problem (g, g^a, g^b, R) randomly, the aim of the challenger $\mathscr{C}$ is to decide if $R = e(g, g)^{a^2 b}$ holds or not. The specific processes of the security game are as follows:

- **Setup.** The challenger $\mathscr{C}$ performs the Setup algorithm and returns the relevant system global parameters to the adversary $\mathscr{A}$as follows:

(1) The challenger $\mathscr{C}$ firstly defines two secure one-way hash functions: $H_0(\omega) = g_1^{-\omega} g^{r_\omega}$, where $r_\omega \in_R Z_q^*$, and $H_1(R) = K$, where $K \in_R \{0, 1\}^k$.

(2) The challenger $\mathscr{C}$ chooses two random numbers $r, v \in Z_q^*$, and returns the system global parameters to the adversary $\mathscr{A}$.

(3) The challenger $\mathscr{C}$ chooses two random numbers $r, v \in Z_q^*$, and returns the system global parameters $PP = (G_1, G_2, H_0, H_1, H_2,\ g, g_1 = g^a, g_2 = g^{ar}, g_3 = g^{vr}, e)$ to the adversary $\mathscr{A}$. The adversary $\mathscr{A}$ can only obtain the value of hash function by performing hash query. And $MSK = \{\alpha = a, \beta = av\}$ is the master secret key. In addition, the challenger $\mathscr{C}$ maintains a list *TKList*, which is initially an empty list.

- **Query Phase I.** The adversary $\mathscr{A}$ initiates a series of queries as follows:

  – *TK* Query: The adversary $\mathscr{A}$ submits an attribute set S to the challenger $\mathscr{C}$. If the adversary $\mathscr{A}$ has been queried the cloud-assisted transformation key TK before, then $\mathscr{C}$ returns the previous TK. Otherwise, the challenger $\mathscr{C}$ returns the cloud-assisted transformation key as follows:

(1) The challenger $\mathscr{C}$picks $d', t, k \in_R Z_q^*$.

(2) The challenger $\mathscr{C}$ randomly chooses a number $u_i \in Z_q^*$ for each attribute $i \in S$, and computes $TK_i = (TK_{i,1}, TK_{i,2})$, where $TK_{i,1} = g_2^{t(\alpha+d)}(g_1^i H_0(\omega_i))^{tu_i}$, $TK_{i,2} = g^{tu_i}$.

(3) The challenger $\mathscr{C}$ returns $TK = (\{TK_i\}_{i\in S}, g^{tv^{-1}d'} g_1^{-tv^{-1}}, g^t, vk, g_1^{tk})$ to the adversary $\mathscr{A}$and adds tuple (TK, UK) to the list $TKList$.

  – *UK* Query. The adversary $\mathscr{A}$ submits an attribute set S to the challenger $\mathscr{C}$. If the adversary $\mathscr{A}$ has been queried for the user decryption key UK before, then $\mathscr{C}$ returns the previous UK, otherwise, the challenger $\mathscr{C}$ makes TK Query to get TK and UK, and finally returns the UK.

  – $Transform_{out}$ Query. In this phase, $\mathscr{A}$ runs the $Transform_{out}(PP, CT, TK) \rightarrow CT'$ algorithm to transform the complicated ciphertext CT to a simplified ciphertext CT' by leveraging the cloud-assisted transformation key TK obtained in TK Query.

- **Challenge.** The adversary $\mathscr{A}$ submits two messages m_0, m_1 with the same length and an access policy tree T^* to the challenger $\mathscr{C}$. The challenger $\mathscr{C}$ randomly picks a bit $coin \in \{0, 1\}$, and responds to the challenging ciphertext CT^* as follows:

1) Let $q_{root}(0) = r^*$, where the node *root* is the root node of T^*, and compute $q_x(0)$ for each leaf node x by using the first step in $Encrypt(PP, T, M) \rightarrow CT$ algorithm of the PVOD-ABE.
2) Compute $C_x^* = (C_{x,1}^*, C_{x,2}^*)$ for each leaf node x, where $C_{x,1}^* = g^{bq_x(0)}$, $C_{x,2}^* = g^{br_x^* q_x(0)}$, and r_x^* can be obtained by making queries $H_0(\omega_x^*)$. Here, $\omega_x^* = att(x)$. Let S^* denote the set of leaf nodes in the access tree T^*.
3) Compute $C_0^* = m_{coin} \oplus H_1(R^{rr^*})$, $C_1^* = (g^b)^{vrr^*} = g_3^{br^*}$.
4) Return $CT^* = (T^*, C_0^*, C_1^*, \{C_x^*\}_{x\in S^*})$ to the adversary $\mathscr{A}$.

Note that the adversary $\mathscr{A}$ cannot simultaneously make the TK Query and the UK Query on the attribute sets that satisfy the challenging access structures T^*.

- **Query Phase II.** The adversary $\mathscr{A}$ still can launch the queries adaptively as **Query Phase I** except for the two queries below:

1) The adversary $\mathscr{A}$ cannot simultaneously make the TK Query and the UK Query on the attribute sets that satisfy the challenging access structures T^*.
2) If the adversary $\mathscr{A}$ has made the TK Query, he/she cannot make the UK Query on the attribute sets that satisfy the challenging access structures T^* and vice versa.

- **Guess.** Finally, the adversary $\mathscr{A}$ outputs a guess $coin'$ for $coin$. If $coin' = coin$ holds, the challenger $\mathscr{C}$ outputs "1", otherwise, outputs "0".

According to the *UserDecrypt* algorithm, it is easy to get $temp^* = e(g_2^a, g^{br^*})$, that is, if $R = e(g,g)^{a^2b}$, the CT^* is the valid ciphertext of m_{coin}, therefore, based on the response from the adversary $\mathscr{A}$, the challenger $\mathscr{C}$ can determine whether $R = e(g,g)^{a^2b}$ holds or not.

Additionally, there is no failure case in the security game, therefore, if $\mathscr{A}$ can break the proposed scheme with non-negligible probability ε, we can conclude that $\mathscr{C}$ can also decide if $R = e(g,g)^{a^2b}$ holds or not with the same non-negligible probability ε. In other words, DBDHP could be solved. However, it is a paradox to the well accepted DBDH problem. Therefore, the advantage of adversary $\mathscr{A}$ breaking the proposed scheme is negligible and our scheme achieves CPA security.

Theorem 6.2. The proposed BC-PC-Share scheme can satisfy the property of completeness.

Proof. According to the traceability and immutability properties of the blockchain, the integrity and consistency of the shared data in our scheme can be guaranteed. During the data sharing process, the deployed smart contracts compare the hash value of ciphertext CT returned by the cloud server in transformation authenticate information *Authen*2 with the hash value retrieved from the hash list pointed by the pointer *link* in blockchain to verify the integrity and consistency of the shared data. If the above two hash values are equal, it can ensure that the returned shared data is consistent with the requested shared data and the integrity of the shared data is not compromised. In addition, the smart contracts verify the validation of the transformed ciphertext by checking whether the equation $temp^{\alpha^{-1}\beta k} = e(C_1, g_1^{tk})$ holds or not. Therefore, our scheme can satisfy the property of completeness.

Theorem 6.3. Under the premise that the cloud server does not collude with the data user DU, the proposed BC-PC-Share scheme can achieve the property of efficient user revocation.

Proof. Assume that the cloud server does not collusion with the data user DU, the cloud server will provide the cloud-assisted transformation service of the ciphertext of the data owner DU within the valid period. In other words, according to the proof of Theorem 6.1, if the data owner DU is not within the valid period, even if the data user DU has made the *UK* Query and obtained the user decryption key *UK*, he/she is forbidden to launch the $Transform_{out}$ Query either, thereby ensuring that the data user DU who is not within the valid period cannot obtain the desired data. In this sense, our scheme realizes the user revocation. In addition, different from most of the existing user revocation schemes, our scheme does not require the system to perform the complicated ciphertext re-encryption and key update operations. Therefore, the proposed BC-PC-Share scheme can achieve the property of efficient key revocation.

7 Performance Evaluation

In this section, we provide the performance evaluation of our proposed BC-PC-Share scheme *vs*. three existing relevant attribute-based data sharing schemes proposed in [19,40,41]. The concrete comparison is divided into three terms. Firstly, we make a comparison of the security properties among the proposed BC-PC-Share scheme and that of schemes in [19,40,41]. Secondly, the complexity analyses in terms of computation cost and storage overhead are presented. Thirdly, we perform the efficiency analysis by a numerical simulation experiment.

7.1 Comparison of Properties

In this subsection, we make a comparison of the security properties among the proposed BC-PC-Share scheme and that of schemes in [19,40,41] in terms of the properties of access control, blockchain-based, sharing consistency, ciphertext transformation, and time-controlled access, and the comparison results are summarized in Table 1. All the schemes in [19,40,41] support the property of the fine-grained access control. In addition, the scheme in [40] can support the properties of the fine-grained access control, blockchain-based, and ciphertext transformation, but fails to support the properties of sharing consistency and time-controlled access. The scheme in [41] can support the properties of fine-grained access control and blockchain-based, but fails to support the properties of sharing consistency, ciphertext transformation, and time-controlled access. The comparison results in Table 1 validate that only the proposed BC-PC-Share scheme satisfies all the desired security features and is more promising and suitable for secure medical data sharing in multiple receiver scenarios.

Table 1: Comparison of security properties

Property	Scheme [19]	Scheme [40]	Scheme [41]	Ours
Access control	✓	✓	✓	✓
Blockchain-based	×	✓	✓	✓
Sharing public verification	×	×	×	✓
Ciphertext transformation	×	✓	×	✓
Time-controlled access	×	×	×	✓

Note: ✓: denotes that the specified scheme satisfy the corresponding property; ×: denotes that the specified scheme does not satisfy the corresponding property.

7.2 *Complexity Analysis*

In this subsection, we perform the comparison in terms of computation cost and communication overhead, respectively. For the sake of the comparison, the notations and their meanings used in this subsection are summarized in Table 2.

Table 2: Notations and their meanings

Notation	Meaning
T_{pair}	The computational cost of one bilinear pairing operation
T_{exp}	The computational cost of one group exponentiation operation in G_1
T_{mul}	The computational cost of one group multiplication operation in G_1
T_{inv}	The computational cost of one group inverse operation in G_1
T_{hash}	The computational cost of one general one-way hash function operation
N_1	The number of attributes involved in the access policy
N_2	The number of attributes involved in the user decryption key
k	The size of hash function
$\|G_1\|/\|G_2\|/\|Z_q\|$	The size of an element of $G_1/G_2/Z_q$

Table 3 gives the comparison of the computation cost between the proposed BC-PC-Share scheme with the existing schemes in [19,40,41] and the comparison of communication overhead among them is briefly presented in Table 4.

Table 3: Comparison of computation cost

Algorithm	Scheme [19]	Scheme [40]	Scheme [41]	Ours
Encrypt	$(12N_1 + 6)T_{exp} + T_{mul} + (N_1 + 3)T_{hash}$	$(3N_1 + 2)T_{exp} + N_1T_{inv} + N_1T_{hash}$	$(2N_1 + 2)T_{exp} + N_1T_{hash}$	$(3N_1 + 2)T_{exp} + N_1T_{hash}$
$Transform_{out}$	$(4N_2 + 2)T_{pair} + \left(2\sum_{z\in T} d_z + 1\right)T_{mul} + 1T_{hash}$	$(2N_2 + 1)T_{pair} + \left(\sum_{z\in T} d_z\right)T_{exp} + 1T_{inv}$	None	$(2N_2 + 1)T_{pair} + \left(\sum_{z\in T} d_z\right)T_{exp} + N_2T_{inv}$
UserDecrypt	$2T_{pair} + 3T_{exp} + 1T_{hash} + 2T_{inv}$	$T_{exp} + T_{inv} + T_{hash}$	$(2N_2 + 1)T_{pair} + \left(\sum_{z\in T} d_z\right)T_{exp} + (N_2 + 1)T_{inv} + T_{hash}$	$1T_{mul}$

Table 4: Comparison of communication overhead

Length	Scheme [19]	Scheme [40]	Scheme [41]	Ours
CT	$(4N_1+3)\lvert G_1\rvert+\lvert G_2\rvert$	$(2N_1+1)\lvert G_1\rvert+\lvert G_2\rvert$	$(2N_1+1)\lvert G_1\rvert+\lvert G_2\rvert$	$(2N_1+1)\lvert G_1\rvert+k$
TK	$(4N_2+2)\lvert G_1\rvert$	$(N_2+2)\lvert G_1\rvert$	None	$(2N_2+3)\lvert G_1\rvert+\lvert Z_q\rvert$
UK	$\lvert Z_q\rvert$	$\lvert Z_q\rvert$	$(2N_2+1)\lvert G_1\rvert$	$\lvert Z_q\rvert$

From Table 3, it can be observed that in the Encrypt phase, the computation cost on the data owner side in our scheme is much smaller than that of the schemes in [19,40]. And our scheme only adds one T_{exp} compared with the scheme in [41]. In the phase of $Transform_{out}$, the computation cost on the cloud server side in our scheme is much smaller than that of the scheme in [19], and is slightly higher than that of the scheme in [40] (i.e., more costly). In addition, as the scheme in [41] does not involve the $Transform_{out}$ algorithm, its computation cost for the transform phase is represented as None. In the phase of UserDecrypt, the computation cost on the data user side of the schemes in [40] and ours is much smaller than that of the schemes in [19,41]. As presented in Table 4, the size of the ciphertext, the transformation key, and the user decryption key in our scheme is relative smaller in overall compared with that of the schemes in [19,40,41].

7.3 *Efficiency Analysis*

The simulation experiment is implemented by a rapid cryptographic prototyping toolkit called Charm [42] with Python, and the operating system is Ubuntu 16.04 64-bit.

Note: considering that the specific implementation details of the blockchain may affect the efficiency of the scheme, but will not affect the security of the scheme, in our experiments, we do not simulate the operation of blockchain nodes. To deploy the scheme in specific blockchain application scenarios is our future research work. Besides, considering that the scheme in [19] is not based on blockchain, here, we only make simulation comparison of the three blockchain-based schemes of [40,41] and ours in terms of encryption time, transformation time and user decryption time.

A comparative study of the execution time of encryption phase, transformation phase, and userdecryption phase among the schemes in [40,41] and ours has been depicted in Figs. 3–5, respectively.

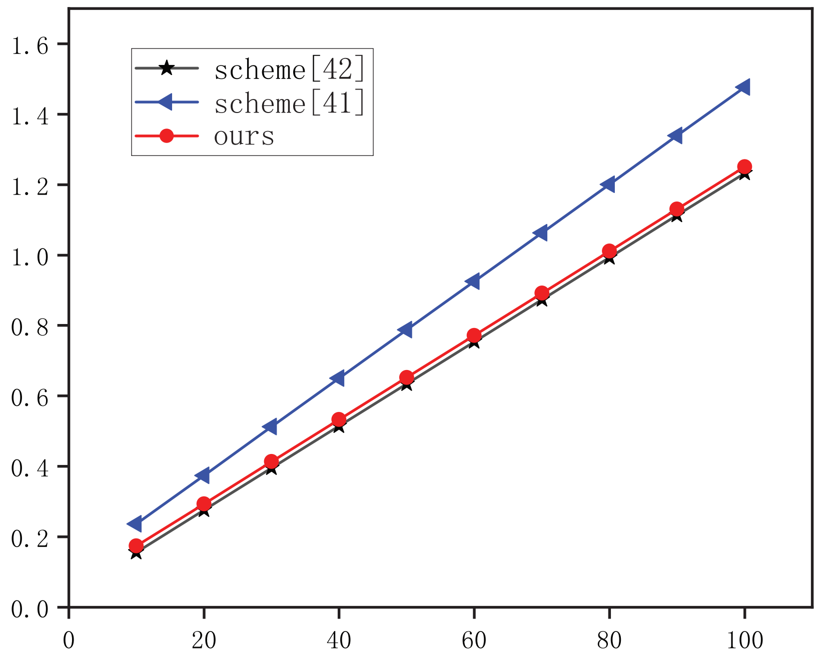

Figure 3: Comparison of encryption time

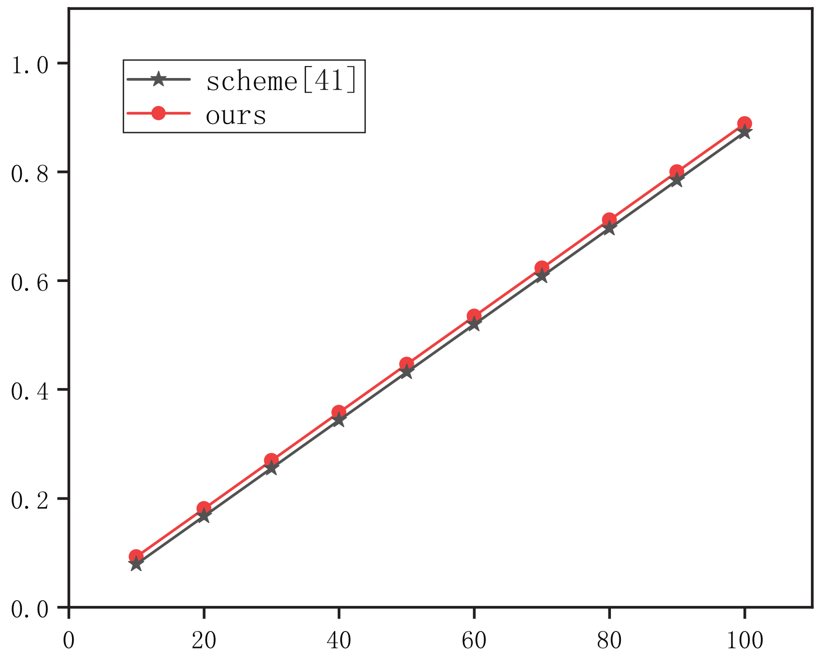

Figure 4: Comparison of transformation time

Fig. 3 illustrates the comparison of computation cost in the encryption phase. The change trend of computation cost in all the three schemes, [40,41] and ours, grows linearly with the number of attributes involved in the access policy. In addition, it is also can be seen that, with the amount of attributes involved in the access policy, the computation cost of scheme [40] is more costly than that of scheme [41] and ours. And the computation cost of scheme [41] is roughly equivalent with ours, and they both keep at a relatively low level.

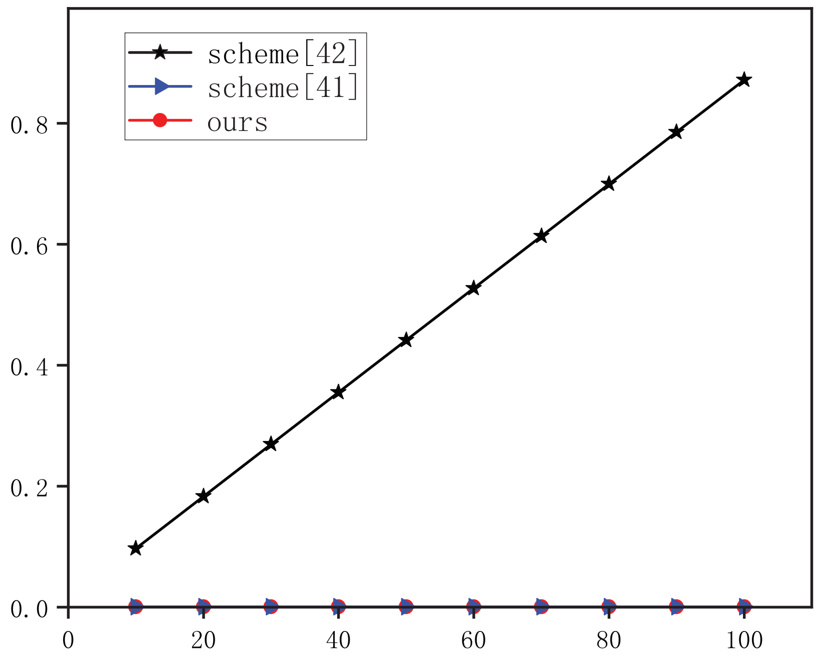

Figure 5: Comparison of userdecryption time

Fig. 4 illustrates the comparison of computation cost in the transformation phase, and it is not surprising to observe that the change trend of computational cost in scheme [41] grows linearly with the amount of attributes involved in the access policy, while it is constant in scheme [40] and ours because the transformation time in scheme [40] and ours does not grow with the number of attributes.

Fig. 5 shows that the change trend of the computation cost in the userdecryption phase, and the change trend of the scheme in [40] and ours grows linearly with the number of attributes and only with a minor gap. Although the computation cost in our scheme is slightly higher than that of the scheme in [40], the former can support the public verification of the integrity and consistency of the shared data, and the latter cannot. Thus, our scheme is more promising and more practical than others.

8 Conclusion

In this paper, we investigated the security issues of PHR data sharing and devised a system paradigm of the combination of cloud computing and consortium blockchain technology and proposed a feasible and promising solution to enhance the trustworthiness of each entity and fulfilled public verifiability, consistency, immutability, scalability and fine-grained PHR sharing in a multi-receiver setting. In addition, our scheme can achieve efficient user revocation and user decryption. The security analysis and simulation experiments conducted in this study demonstrated the security, scalability, and efficiency of the proposed BC-PC-Share scheme.

Acknowledgement: The authors are very grateful to the editors and reviewers for their constructive feedback and insightful suggestions.

Funding Statement: This work was supported by the Youth Doctoral Foundation of Gansu Education Committee under Grant No. 2022QB-176.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. Tang, P. C., Ash, J. S., Bates, D. W., Overhage, J. M., Sands, D. Z. (2006). Personal health records: Definitions, benefits, and strategies for overcoming barriers to adoption. *Journal of the American Medical Informatics Association, 13(2),* 121–126. https://doi.org/10.1197/jamia.M2025
2. Hylock, R. H., Zeng, X. (2019). A blockchain framework for patient-centered health records and exchange (healthchain): Evaluation and proof-of-concept study. *Journal of Medical Internet Research, 21(8),* e13592. https://doi.org/10.2196/13592
3. Mell, P., Grance, T. (2011). The NIST definition of cloud computing. http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf
4. Li, M., Yu, S., Zheng, Y., Ren, K., Lou, W. (2012). Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption. *IEEE Transactions on Parallel and Distributed Systems, 24(1),* 131–143. https://doi.org/10.1109/TPDS.2012.97
5. Au, M. H., Yuen, T. H., Liu, J. K., Susilo, W., Huang, X. et al. (2017). A general framework for secure sharing of personal health records in cloud system. *Journal of Computer and System Sciences, 90(11),* 46–62. https://doi.org/10.1016/j.jcss.2017.03.002
6. Zhang, W., Wu, Y., Xiong, H., Qin, Z. (2021). Accountable attribute-based encryption with public auditing and user revocation in the personal health record system. *KSII Transactions on Internet and Information Systems (TIIS), 15(1),* 302–322.
7. Xhafa, F., Feng, J., Zhang, Y., Chen, X., Li, J. (2015). Privacy-aware attribute-based phr sharing with user accountability in cloud computing. *The Journal of Supercomputing, 71(5),* 1607–1619. https://doi.org/10.1007/s11227-014-1253-3
8. Chen, T. L., Liao, Y. T., Chang, Y. F., Hwang, J. H. (2016). Security approach to controlling access to personal health records in healthcare service. *Security and Communication Networks, 9(7),* 652–666. https://doi.org/10.1002/sec.1387
9. Chen, B., He, D., Kumar, N., Wang, H., Choo, K. K. R. (2021). A blockchain-based proxy re-encryption with equality test for vehicular communication systems. *IEEE Transactions on Network Science and Engineering, 8(3),* 2048–2059. https://doi.org/10.1109/TNSE.2020.2999551
10. Lin, C., He, D., Huang, X., Kumar, N., Choo, K. K. R. (2021). Bcppa: A blockchain-based conditional privacy-preserving authentication protocol for vehicular ad hoc networks. *IEEE Transactions on Intelligent Transportation Systems, 22(12),* 7408–7420. https://doi.org/10.1109/TITS.2020.3002096
11. Chen, L., Lee, W. K., Chang, C. C., Choo, K. K. R., Zhang, N. (2019). Blockchain based searchable encryption for electronic health record sharing. *Future Generation Computer Systems, 95(3),* 420–429. https://doi.org/10.1016/j.future.2019.01.018
12. Shi, S., He, D., Li, L., Kumar, N., Khan, M. K. et al. (2020). Applications of blockchain in ensuring the security and privacy of electronic health record systems: A survey. *Computers & Security, 97(5),* 101966. https://doi.org/10.1016/j.cose.2020.101966
13. McGhin, T., Choo, K. K. R., Liu, C. Z., He, D. (2019). Blockchain in healthcare applications: Research challenges and opportunities. *Journal of Network and Computer Applications, 135(1),* 62–75. https://doi.org/10.1016/j.jnca.2019.02.027
14. Wang, H., Song, Y. (2018). Secure cloud-based EHR system using attribute-based cryptosystem and blockchain. *Journal of Medical Systems, 42(8),* 152. https://doi.org/10.1007/s10916-018-0994-6

15. Xia, Q., Sifah, E. B., Smahi, A., Amofa, S., Zhang, X. (2017). BBDS: Blockchain-based data sharing for electronic medical records in cloud environments. *Information, 8(2),* 44. https://doi.org/10.3390/info8020044
16. Wang, H., Wang, Q., He, D. (2021). Blockchain-based private provable data possession. *IEEE Transactions on Dependable and Secure Computing, 18(5),* 2379–2389.
17. Zhu, L., Wu, Y., Gai, K., Choo, K. K. R. (2019). Controllable and trustworthy blockchain-based cloud data management. *Future Generation Computer Systems, 91(99),* 527–535. https://doi.org/10.1016/j.future.2018.09.019
18. Zhang, Y., Deng, R. H., Liu, X., Zheng, D. (2018). Blockchain based efficient and robust fair payment for outsourcing services in cloud computing. *Information Sciences, 462(4),* 262–277. https://doi.org/10.1016/j.ins.2018.06.018
19. Xiong, H., Zhang, H., Sun, J. (2018). Attribute-based privacy-preserving data sharing for dynamic groups in cloud computing. *IEEE Systems Journal, 13(3),* 2739–2750. https://doi.org/10.1109/JSYST.2018.2865221
20. Li, H., Yu, K., Liu, B., Feng, C., Qin, Z. et al. (2022). An efficient ciphertext-policy weighted attribute-based encryption for the internet of health things. *IEEE Journal of Biomedical and Health Informatics, 26(5),* 1949–1960. https://doi.org/10.1109/JBHI.2021.3075995
21. Bao, Y., Qiu, W., Cheng, X. (2022). Secure and lightweight fine-grained searchable data sharing for IoT-oriented and cloud-assisted smart healthcare system. *IEEE Internet of Things Journal, 9(4),* 2513–2526. https://doi.org/10.1109/JIOT.2021.3063846
22. Li, H., Lan, C., Fu, X., Wang, C., Li, F. et al. (2020). A secure and lightweight fine-grained data sharing scheme for mobile cloud computing. *Sensors, 20(17),* 4720. https://doi.org/10.3390/s20174720
23. Zhang, A., Lin, X. (2018). Towards secure and privacy-preserving data sharing in e-health systems via consortium blockchain. *Journal of Medical Systems, 42(8),* 1–18. https://doi.org/10.1007/s10916-018-0995-5
24. Thwin, T. T., Vasupongayya, S. (2019). Blockchain-based access control model to preserve privacy for personal health record systems. *Security and Communication Networks, 2019(5),* 8315614. https://doi.org/10.1155/2019/8315614
25. Cao, S., Zhang, G., Liu, P., Zhang, X., Neri, F. (2019). Cloud-assisted secure eHealth systems for tamper-proofing EHR via blockchain. *Information Sciences, 485(3),* 427–440. https://doi.org/10.1016/j.ins.2019.02.038
26. Nagasubramanian, G., Sakthivel, R. K., Patan, R., Gandomi, A. H., Sankayya, M. et al. (2020). Securing e-health records using keyless signature infrastructure blockchain technology in the cloud. *Neural Computing and Applications, 32(3),* 639–647. https://doi.org/10.1007/s00521-018-3915-1
27. Yang, Z., Zhang, H., Yu, H., Li, Z., Zhu, B. et al. (2021). Attribute-based keyword search over the encrypted blockchain. *Computer Modeling in Engineering & Sciences, 128(1),* 269–282. https://doi.org/10.32604/cmes.2021.015210
28. Al Omar, A., Bhuiyan, M. Z. A., Basu, A., Kiyomoto, S., Rahman, M. S. (2019). Privacy-friendly platform for healthcare data in cloud based on blockchain environment. *Future Generation Computer Systems, 95(10),* 511–521. https://doi.org/10.1016/j.future.2018.12.044
29. Lu, Y. (2019). The blockchain: State-of-the-art and research challenges. *Journal of Industrial Information Integration, 15(4),* 80–90. https://doi.org/10.1016/j.jii.2019.04.002
30. Lin, C., He, D., Huang, X., Khan, M. K., Choo, K. K. R. (2020). DCAP: A secure and efficient decentralized conditional anonymous payment system based on blockchain. *IEEE Transactions on Information Forensics and Security, 15,* 2440–2452. https://doi.org/10.1109/TIFS.2020.2969565
31. Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. https://bitcoin.org/bitcoin.pdf
32. Khan, S. N., Loukil, F., Ghedira-Guegan, C., Benkhelifa, E., Bani-Hani, A. (2021). Blockchain smart contracts: Applications, challenges, and future trends. *Peer-to-Peer Networking and Applications, 14(5),* 2901–2925. https://doi.org/10.1007/s12083-021-01127-0

33. Bethencourt, J., Sahai, A., Waters, B. (2007). Ciphertext-policy attribute-based encryption. *2007 IEEE Symposium on Security and Privacy (SP'07)*, pp. 321–334. Berkeley, CA, USA, IEEE. https://doi.org/10.1109/SP.2007.11
34. Sahai, A., Waters, B. (2005). Fuzzy identity-based encryption. In: Cramer, R. (Ed.) *Lecture notes in computer science*, vol. 3494. Berlin, Heidelberg: Springer. https://doi.org/10.1007/11426639_27
35. Goyal, V., Pandey, O., Sahai, A., Waters, B. (2006). Attribute-based encryption for fine-grained access control of encrypted data. *Proceedings of the 13th ACM Conference on Computer and Communications Security (CCS'06)*. pp. 89–98. New York, NY, USA, Association for Computing Machinery. https://doi.org/10.1145/1180405.1180418
36. Yang, J. J., Li, J. Q., Niu, Y. (2015). A hybrid solution for privacy preserving medical data sharing in the cloud environment. *Future Generation Computer Systems, 43(8),* 74–86. https://doi.org/10.1016/j.future.2014.06.004
37. Zhang, Y., He, D., Choo, K. K. R. (2018). BaDS: Blockchain-based architecture for data sharing with ABS and CP-ABE in IoT. *Wireless Communications and Mobile Computing, 2018(2),* 2783658. https://doi.org/10.1155/2018/2783658
38. Xu, J., Xue, K., Li, S., Tian, H., Hong, J. et al. (2019). Healthchain: A blockchain-based privacy preserving scheme for large-scale health data. *IEEE Internet of Things Journal, 6(5),* 8770–8781. https://doi.org/10.1109/JIOT.2019.2923525
39. Lai, J., Deng, R. H., Guan, C., Weng, J. (2013). Attribute-based encryption with verifiable outsourced decryption. *IEEE Transactions on Information Forensics and Security, 8(8),* 1343–1354. https://doi.org/10.1109/TIFS.2013.2271848
40. Zheng, H., Shao, J., Wei, G. (2020). Attribute-based encryption with outsourced decryption in blockchain. *Peer-to-Peer Networking and Applications, 13(5),* 1643–1655. https://doi.org/10.1007/s12083-020-00918-1
41. Ezhil Arasi, V., Indra Gandhi, K., Kulothungan, K. (2022). Auditable attribute-based data access control using blockchain in cloud storage. *The Journal of Supercomputing, 78(8),* 10772–10798. https://doi.org/10.1007/s11227-021-04293-3
42. Akinyele, J. A., Garman, C., Miers, I., Pagano, M. W., Rushanan, M. et al. (2013). Charm: A framework for rapidly prototyping cryptosystems. *Journal of Cryptographic Engineering, 3(2),* 111–128. https://doi.org/10.1007/s13389-013-0057-3