electronics

MDPI

Article

Privacy-Preserving Fine-Grained Redaction with Policy Fuzzy Matching in Blockchain-Based Mobile Crowdsensing

Hongchen Guo [1], Haotian Liang [2], Mingyang Zhao [2], Yao Xiao [2], Tong Wu [2,*], Jingfeng Xue [1] and Liehuang Zhu [2]

[1] School of Computer Science and Technology, Beijing Institute of Technology, Beijing 100081, China; guohongchen@bit.edu.cn (H.G.); xuejf@bit.edu.cn (J.X.)
[2] School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing 100081, China; haotianl@bit.edu.cn (H.L.); mingyangz@bit.edu.cn (M.Z.); 3120195529@bit.edu.cn (Y.X.); liehuangz@bit.edu.cn (L.Z.)
* Correspondence: tracy_tongw@163.com

Abstract: The redactable blockchain has emerged as a promising technique in mobile crowdsensing, allowing users to break immutability in a controlled manner selectively. Unfortunately, current fine-grained redactable blockchains suffer two significant limitations in terms of security and functionality, which severely impede their application in mobile crowdsensing. For security, the transparency of the blockchain allows anyone to access both the data and policy, which consequently results in a breach of user privacy. Regarding functionality, current solutions cannot support error tolerance during policy matching, thereby limiting their applicability in various situations, such as fingerprint-based and face-based identification scenarios. This paper presents a privacy-preserving fine-grained redactable blockchain with policy fuzzy matching, named PRBFM. PRBFM supports fuzzy policy matching and partitions users' privileges without compromising user privacy. The idea of PRBFM is to leverage threshold linear secret sharing based on the Lagrange interpolation theorem to distribute the decryption keys and chameleon hash trapdoors. Additionally, we have incorporated a privacy-preserving policy matching delegation mechanism into PRBFM to minimize user overhead. Our security analysis demonstrates that PRBFM can defend against the chosen-ciphertext attack. Moreover, experiments conducted on the FISCO blockchain platform show that PRBFM is at least 7.8 times faster than existing state-of-the-art solutions.

Keywords: blockchain; fine-grained redaction; policy fuzzy matching; privacy preservation; mobile crowdsensing

Citation: Guo, H.; Liang, H.; Zhao, M.; Xiao, Y.; Wu, T.; Xue, J.; Zhu, L. Privacy-Preserving Fine-Grained Redaction with Policy Fuzzy Matching in Blockchain-Based Mobile Crowdsensing. *Electronics* **2023**, *12*, 3416. https://doi.org/10.3390/electronics12163416

Academic Editor: Seokjoo Shin

Received: 16 July 2023
Revised: 7 August 2023
Accepted: 8 August 2023
Published: 11 August 2023

1. Introduction

With the increasing prevalence of intelligent terminals, particularly in light of contemporary trends such as "Industrie 4.0" [1] and the IoT, mobile crowdsensing has emerged as a promising application that leverages smart devices in mobile networks to utilize idle resources for sensing tasks effectively [2]. However, the current mobile crowdsensing paradigms primarily rely on centralized platforms that are not entirely trustworthy in practice and give rise to issues such as fraud, security vulnerabilities, and the single point of failure [3]. Consequently, the adoption of blockchain techniques has become widespread as a means to address these challenges. In essence, the immutability of blockchain serves as a critical measure against any manipulation of registered objects for illicit gains [4,5].

However, the immutability of blockchain may impede the application of the blockchain. Nowadays, the blockchain ecosystem is plagued by the presence of inappropriate materials, such as fake news, copyrighted content, and sensitive data [6–9]. Researchers have discovered that individuals with malicious intent can upload objectionable material (such as malware or pornographic links) onto the Bitcoin blockchain [7]. Regrettably, no effective methods currently exist to prevent the dissemination of this harmful content throughout the Bitcoin network. Participants on the chain may be concerned about being associated

with illegal issues, discouraging their involvement in and use of the chain. Furthermore, certain blockchain systems have faced the issue of illicit transactions in practical scenarios. For instance, a well-known incident called the "The DAO" attack took place in Ethereum due to the presence of susceptible smart contracts [9]. This attack even necessitated a hard fork to mitigate the resulting adverse consequences. Additionally, various data regulations and requirements, such as the General Data Protection Regulation (GDPR) [10] and the concept of "the right to be forgotten" [11], empower individuals to manage their personal data [12–14]. However, because of the immutability of blockchain, it is evident that the information stored on the chain cannot be altered [15].

In order to break the immutability of the blockchain in a controlled manner selectively, the idea of the redactable blockchain was introduced by Ateniese et al. [16]. However, their method is on the basis of the policy-based chameleon hash (PCH) [6] that inherits the limitations of traditional attribute-based encryption (ABE) [17]. To address this problem, Tian et al. [18] proposed a novel implementation of attribute-based traitor tracing (ABTT) [19,20] that utilizes Hierarchical Identity-Based Encryption (HIBE) and one-time signature schemes. However, this proposed scheme only provides limited accountability. Therefore, Xu et al. [21] proposed a novel scheme of PCH with black-box accountability (PCHA) to overcome this challenge. Moreover, to satisfy the growing need in real-world scenarios, various redactable blockchain schemes have been proposed with necessity characters: dynamism [22], verifiability [23], and flexibility [24].

Unfortunately, these aforementioned redactable blockchain schemes all fall short in the aspects of security and functionality. For security, the data and policy can be accessed by any entity because of the transparency of the blockchain, which may result in potential privacy leakage. When it comes to functionality, existing works lack support for error tolerance during policy matching [25]. For instance, in some real-world scenarios such as fingerprint identification [26] or face recognition [27], it is almost impossible to achieve a 100% perfect match. To sum up, there is an urgent need for a fine-grained redactable blockchain with policy fuzzy matching in a privacy-preserving manner. Hence, four challenges arise:

1. How to enable fine-grained redactable blockchain with fuzzy policy matching;
2. How to conceal the policy based on fuzzy policy matching;
3. How to ensure data privacy while maintaining privilege downward compatibility (i.e., allowing redactable users to access the data) through policy concealment;
4. How to minimize user overhead in a privacy-preserving paradigm.

1.1. Contribution

In this paper, we provide a positive answer to the aforementioned problems by proposing a privacy-preserving fine-grained redactable blockchain with fuzzy policy matching (PRBFM). Specifically, PRBFM supports fuzzy policy matching and partitions users' privileges without compromising user privacy. The main contributions are illustrated below.

- We introduce a novel privacy-preserving fine-grained redactable blockchain with fuzzy policy matching for mobile crowdsensing scenarios. Concretely, to achieve data privacy preservation and fuzzy matching for the redactable blockchain, we leverage the Lagrange interpolation theorem-based secret sharing to distribute the data decryption keys and chameleon hash trapdoors.
- To further satisfy the requirement of privacy-preserving policy matching and reduce the user overhead, we design a privacy-preserving policy matching delegation mechanism for PRBFM.
- A formal security analysis is provided to demonstrate the security of PRBFM against chosen-ciphertext attacks in a random oracle model. Subsequently, we employ a real-world dataset to perform experiments on the FISCO blockchain platform. The experimental results demonstrate that our schemes outperform related existing solutions, with a speed improvement of a minimum of 7.8×.

1.2. Roadmap

The structure of this paper is shown in Figure 1. Section 2 details the system model, threat model, and problem formulation. Proceeding further, Section 3 presents the preliminaries utilized in PRBFM. Subsequently, in Section 4, we formalize our scheme by presenting the definition and detailed construction of PRBFM, while the analysis and applications are discussed in Section 5. Several experiments based on the FISCO blockchain and the real-world dataset are conducted in Section 6. Finally, we discuss several related works about the redactable blockchain in Section 7 before reaching the conclusion in Section 8.

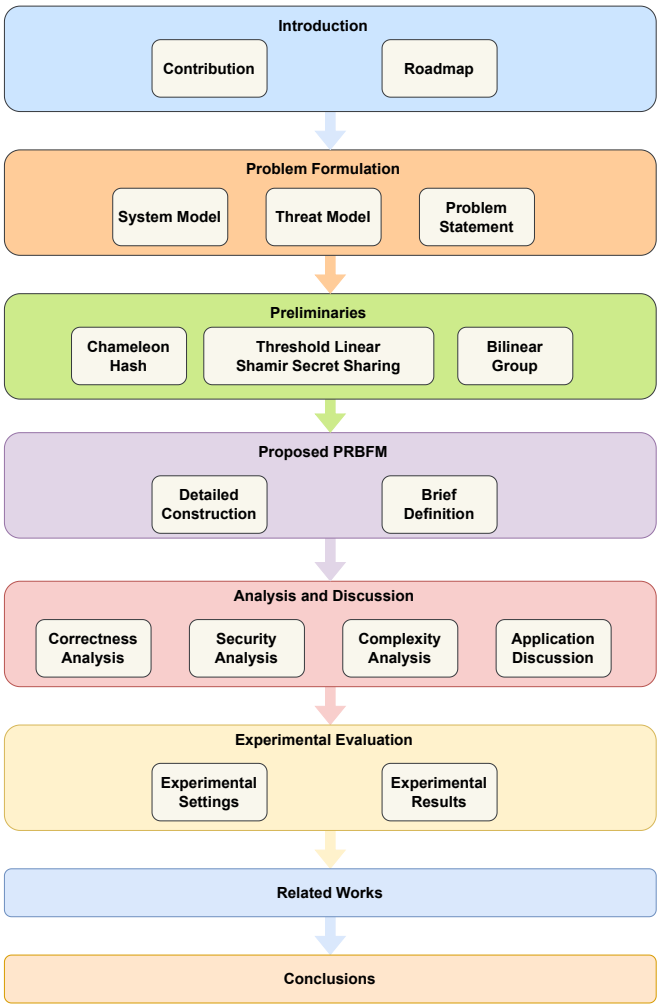

Figure 1. The structure of this paper.

2. Problem Formulation

2.1. System Model

In this section, we formulate the problem addressed by the proposed PRBFM scheme. This scheme involves three different types of entities: *nodes*, *users*, and *data owners*. The nodes are responsible for storing the data of corresponding owners and executing data queries from users with permission. Notably, the nodes can be divided into three main types: privilege nodes (i.e., consortium nodes), storage nodes (i.e., cloud nodes), and computation nodes (i.e., edge nodes). The users usually represent requesters in realistic mobile crowdsensing scenarios. According to their attributes and the policy of the data owner, they can be split into three roles: unauthorized users, authorized users, and authorized modifiers. Unauthorized users are unable to perform any actions on the uploaded data. Authorized users have read-only access to the uploaded data. On the other hand, authorized modifiers can both access and modify the contents of the uploaded data. Similar

to the users, the data owners mainly represent individuals or organizations in real-world mobile crowdsensing scenarios. Their prior task is to crowdsense raw data, then encrypt and upload them to the blockchain. The system model is presented in Figure 2.

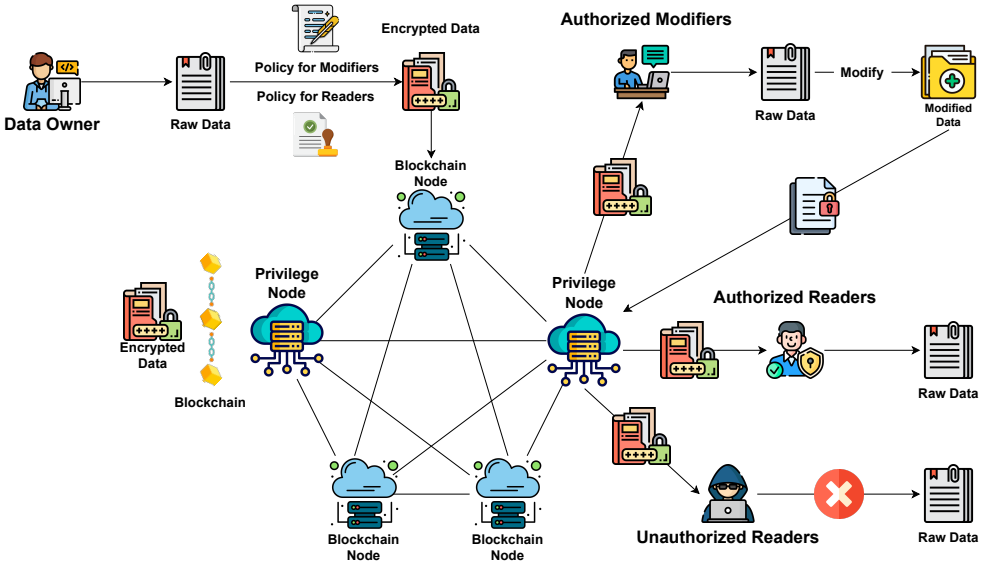

Figure 2. System model of our scheme.

2.2. Threat Model

In blockchain-based mobile crowdsensing scenarios, it is assumed that the privileged nodes, such as the government, organization administrators, and manufacturers, are trustworthy entities as they regulate the blockchain and authorize user privileges [28–30]. Conversely, other entities such as users, data owners, storage nodes, and computation nodes are semi-honest. Though they are expected to follow protocols faithfully, they have the curiosity to probe others' private data, which may lead to unwanted inferences [7,31]. Additionally, users may maliciously collude with other entities except for the privileged nodes, resulting in attacks such as privilege escalation and privacy eavesdropping. We summarize the potential attacks below.

- **Unauthorized access attack:** Since encrypted data submitted by corresponding owners may contain commercially sensitive or private details, they become a prime target of potential adversaries. One such threat is the unauthorized access attack, whereby individuals without authorization may attempt to read or modify the data.
- **Eavesdropping attack:** An unauthorized party may eavesdrop on data transmitted through public channels in an attempt to deduce sensitive details from the intercepted ciphertext.
- **User inferring attack:** Attributes and policies may contain specific characteristics of individuals, such as their occupation, rank, or identity. An attack that expert adversaries may use to elicit sensitive information without knowing personal identifiers, such as the user identity or type of encrypted data, is known as the user inferring attack.
- **Identity disguising attack:** Access to encrypted data is determined based on specific attributes of individual users, including their identity. As a result, identity disguising attacks occur when unauthorized entities attempt to appear as authorized readers or modifiers. These attacks may involve disguising the encryption key with unauthorized attributes or attaching unauthorized attributes to obtained ciphertexts to mislead other entities. In some instances, unauthorized actors may alter or tamper with the message to deceive other parties.
- **Collusion attack:** Unauthorized entities may collaborate to perform various attacks, including those outlined above. For example, unauthorized users may pool their secret keys to recover encrypted data without proper authorization. They may also ex-

change keys to obtain encrypted data without proper authorization from the specified sender's attributes.

2.3. Problem Statement and Design Goals

The problem addressed in this paper is as follows: a data owner encrypts plaintext message m based on a readability policy $\mathcal{P}r$ and an editability policy $\mathcal{P}e(\mathcal{P}r \subset \mathcal{P}e)$. The resulting encryption (C, H) is then submitted to blockchain nodes. Simultaneously, a user with attributes σ exists. The design goal of PRBFM is to *achieve a secure redactable blockchain in which the user can read m only when* $\mathcal{P}r \subset \sigma$ *and can edit only if* $\mathcal{P}e \subset \sigma$ *while maintaining the confidentiality of sensitive data, including data from owners, user attributes, and owner policies, to avoid exposure to other parties.*

3. Preliminaries

3.1. Chameleon Hash

Ateniese et al. [16] proposed the chameleon hash. A chameleon hash system typically comprises five algorithms that can be executed in polynomial time. These definitions are illustrated below:

- **PPGen(1^λ)**. This algorithm's input is the security parameter λ. The output of this algorithm is the public parameters PP. Notably, we implicitly assume that mpk is the input for the subsequent algorithms.
- **KeyGen(PP)**. This algorithm uses the public parameters PP as input and generates the public key pk and secret key sk as output.
- **HashGen(pk, m)**. This algorithm takes the public key pk and plaintext message m as input, and outputs the hash h and random value r.
- **Verify(pk, m, r, h)**. This algorithm takes the public key pk, plaintext message m, random value r, and hash h as input, and generates the decision result d as output. Specifically, d is equal to 1 if the hash is valid, and 0 otherwise.
- **Adapt(sk, m, m', r, h)**. This algorithm takes the secret key sk, plaintext message m, alternate message m', random value r, and hash h as input, and generates the alternate random value r' as output.

We make the assumption, without loss of generality, that the $Adapt$ algorithm always requires the hash h for verification. If h is invalid, the algorithm outputs $\perp$ instead of the alternate random value r'.

3.2. Threshold Linear Shamir Secret Sharing

In [32], Shamir proposed a Lagrange interpolation theorem-based secret sharing scheme. Given k points $(x_1, y_1), \cdots, (x_k, y_k)$ on the 2-D plane where the values of x_i are distinct, there is one and only one interpolation polynomial $q(x)$ of degree $k-1$ that satisfies:

$$y_i = q(x_i), \forall i = 1, 2, \cdots, k. \tag{1}$$

Without loss of generality, we can assume that the confidential data D are a number. To divide D into n shares, denoted as D_i, $i = 1, 2, \cdots, n$, a random polynomial $q(x)$ of degree $k-1$ is selected as:

$$q(x) = a_0 + a_1 x + \cdots + a_{k-1} x_{k-1}, \tag{2}$$

where $a_0 = D$. For $i = 1, 2, \cdots, n$, the corresponding piece D_i is computed as $D_i = q(i)$.

Using the Lagrange interpolation technique, the coefficient of f can be computed when given any subset of k shares from D_i. The secret data D can be obtained by calculating $D = q(0)$. Note that if we have fewer than k shares, D cannot be reconstructed. Specifically, having only $k-1$ or fewer shares does not reveal any information about D.

3.3. Bilinear Group

Definition 1. *(Bilinear Group). Suppose $\mathbb{G}_1, \mathbb{G}_2$, and $\mathbb{G}_T$ are three bilinear groups. There exists a binlinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, where $|\mathbb{G}_1| = |\mathbb{G}_2| = |\mathbb{G}_T| = p$.*

Let these three groups have the same prime order p. The generators of $\mathbb{G}_1$ and $\mathbb{G}_2$ are g and h, respectively. The map e has the following two properties: Binlinearity and Non-degeneration.

If $\mathbb{G}_1 = \mathbb{G}_2$, it is called symmetric pairing. Otherwise, two different types of asymmetric pairing exist based on the existence of the isomorphism function that from $\mathbb{G}_2$ to $\mathbb{G}_1$.

Then, we present the computationally intractable problem that is utilized in this paper.

Definition 2. *(Decisional Modified Bilinear Diffie-Hellman Problem (MBDFH). Let g_i^a, g_j^b, g_k^c and $e(g_1, g_2)^z$ evaluate whether $e(g_1, g_2)^{\frac{ab}{c}} = e(g_1, g_2)^z$, where $i, j, k \in \{1, 2\}$. Suppose there exists an algorithm that generates the group as $\mathcal{G}$, the distribution D is defined as follows:*

$$\begin{aligned} \mathbb{G} &\stackrel{def}{=} (q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e) \leftarrow \mathcal{G}, a, b, c, z \in \mathbb{Z}_q, \\ D &\stackrel{def}{=} (G; g_1, g_2, g_i^a, g_j^b, g_k^c, i, j, k \in \{1, 2\}). \end{aligned} \tag{3}$$

Suppose $\mathcal{A}$ represents a probabilistic polynomial-time (PPT) adversary breaking the MBDH problem.

$$Adv_{\mathcal{A}}^{MBDH}(\lambda) \stackrel{def}{=} |Pr[\mathcal{A}(D, e(g_1, g_2)^{\frac{ab}{c}})] - Pr[\mathcal{A}(D, e(g_1, g_2)^z)]| \tag{4}$$

is negligible for the security parameter λ.

4. Proposed PRBFM

The primary objective of PRBFM is to utilize two techniques. The first technique involves employing Lagrange secret sharing and a chameleon hash, which allow us to create a redactable blockchain having privilege downward compatibility and fuzzy matching while still maintaining privacy. Meanwhile, the second technique involves designing a privacy-preserving matching delegation mechanism, which minimizes user overhead.

4.1. Brief Definition

Before illustrating the details of PRBFM, we will first provide a brief definition:

Definition 3. *PRBFM comprises eight polynomial-time algorithms:* **Setup**, **KeyGen**, **Encrypt**, **Verify**, **TrGen**, **Match**, **Read**, *and* **Edit**.

- **Setup ($\lambda \to (mpk, msk)$)**. *The algorithm yields the master secret key msk and master public key mpk when given a security parameter λ. For simplicity, mpk is implicitly assumed to be taken as input by all other algorithms.*
- **KeyGen ($(msk, \sigma) \to sk$)**. *The algorithm takes σ and msk as inputs and produces the user's secret key sk.*
- **Encrypt ($(m, x, \mathcal{P}_r, \mathcal{P}_e) \to (C, H)$)**. *The algorithm takes message $m \in \mathcal{M}$, Chameleon hash trapdoor x, readability policy $\mathcal{P}_r$, and editability policy $\mathcal{P}_e (\mathcal{P}_r \subset \mathcal{P}_e)$ as inputs and outputs ciphertext C along with the corresponding hash value $H = (h, r_h)$.*
- **Verify ($(C, H) \to d_v$)**. *The algorithm produces a verification result d_v by validating the pair (C, H), with $d_v \in 0, 1$.*
- **TrGen ($sk \to T$)**. *The algorithm takes the secret key sk as input to generate the trapdoor T, which is composed of T_1 and T_{2,i_i^σ}.*
- **Match ($(T, C) \to d_m$)**. *Given the trapdoor T and the ciphertext C, the algorithm outputs a match result $d_m \in 0, 1, 2$ to indicate different levels of access. Specifically, (C^1, H) is returned when $d_m = 1$, and (C^2, H) is returned when $d_m = 2$.*

- **Read (**$(C^1, sk) \rightarrow m | \perp$**)**. *Given the ciphertext* C^1 *and the secret key sk as inputs, the algorithm retrieves the message m only if* $\sigma \subset \mathcal{P}_r$*. Otherwise, the algorithm returns an error symbol* $\perp$*.*
- **Edit (**$(C^2, sk, m^{'}, \mathcal{P}_r^{'}, \mathcal{P}_e^{'}) \rightarrow (C^{'}, H^{'})$**)**. *Given the ciphertext* C^2*, secret key sk, and new message* $m^{'}$ *along with their respective policies* $(\mathcal{P}_r^{'}, \mathcal{P}_e^{'})$*, the algorithm generates a new ciphertext* $C^{'}$ *and hash value* $H^{'} = (h, r_h^{'})$*. Importantly, the editability feature preserves the correspondence of on-chain hashes and off-chain data.*

4.2. Detailed Construction

We now describe the detail of PRBFM.

Setup $\lambda \rightarrow (mpk, msk)$: Based on the decentralized key generation protocol, multiple privilege nodes cooperate to generate the system parameters.

- Generate the description of bilinear map $\Gamma = (p, g, \mathbb{G}, \mathbb{G}_T, e)$, where $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$. Subsequently, assume the attribute universe as $\mathcal{U}$ and the size of $\mathcal{U}$ as n. Set the threshold of policy matching as d. Next, generate n random values $\{r_i\}^n$. Then, compute $\{R_i = g^{r_i}\}^n$. Next, select a random value $\alpha \in \mathbb{Z}_p$ and a hash function $H[\cdot] : \mathbb{G}_T \rightarrow \{0,1\}^*$.
- Generate the master secret key $msk = (\{r_i\}^n, \alpha)$ and the master public key $mpk = (\{R_i\}^n, g^{\alpha}, H, \Gamma)$.

KeyGen $(msk, \sigma) \rightarrow sk$: A user selects a privilege node for registration, and the privilege node combines the user's attribute set to generate the secret key.

- Randomly generate a $(d-1)$-degree polynomial $q(x) = \alpha + a_1 x + a_2 x^2 + \cdots + a_{d-1} x^{d-1}$, where $q(0) = \alpha$. Then, for each attribute $i \in \sigma$, compute $q(i)$. Subsequently, utilize the Lagrange interpolation theorem to compute the Lagrange parameters $\{\Delta_{i,\sigma}(0)\}_i^{\sigma}$.
- Generate the secret key $sk = \{sk_i = g^{\frac{q(i)\Delta_{i,\sigma}(0)}{r_i}}\}_i^{\sigma}$. Then, return sk to the user.

Encrypt $(m, x, \mathcal{P}_r, \mathcal{P}_e) \rightarrow (C, H)$: based on readability and editability policies, a user generates the ciphertext and hash value.

- Select two random values $d_1 \in \mathbb{Z}_p$ and $d_2 \in \mathbb{Z}_p$. Generate the readability policy $\mathcal{P}_r$. Compute $\{C_{1,i} = R_i^{d_1}\}_i^{\mathcal{P}_r}$ and $\{C_{2,i} = R_i^{d_2}\}_i^{\mathcal{P}_r}$. Subsequently, compute $C_3 = g^{\alpha d_2}$ and $C_m = m \oplus H[e(g,g)^{\alpha d_1}]$.
- Select two random values $d_3 \in \mathbb{Z}_p$ and $d_4 \in \mathbb{Z}_p$, and the chameleon secret key x. Generate the editability policy $\mathcal{P}_e$. Subsequently, compute $\{C_{4,i} = R_i^{d_3}\}_i^{\mathcal{P}_e}$, $\{C_{5,i} = R_i^{d_4}\}_i^{\mathcal{P}_e}$, $C_6 = g^{\alpha d_4}$, and $C_7 = g^x$. To guarantee the right downward compatibility, compute $C_x = x \oplus H[e(g,g)^{\alpha d_1}] \oplus H[e(g,g)^{\alpha d_3}]$.
- Select a random value $r_h \in \mathbb{Z}_p$ to compute the chameleon hash $h = g^{C_m} g^{x r_h}$.
- Generate the ciphertext $C = (\{C_{1,i}\}_i^{\mathcal{P}_r}, \{C_{2,i}\}_i^{\mathcal{P}_r}, C_3, \{C_{4,i}\}_i^{\mathcal{P}_e}, \{C_{5,i}\}_i^{\mathcal{P}_e}, C_6, C_7, C_m, C_x)$ and hash value $H = (h, r_h)$.
- Send the pair (C, H) to computation nodes and the chameleon hash h to the blockchain. Next, computation nodes upload the pairs to storage nodes.

Verify $(C, H) \rightarrow d_v$: computation nodes verify the validity of the pair (C, H) and output a verification result.

- Check the equation $h \stackrel{?}{=} g^{C_m} C_7^{r_h}$. If the equation holds, output d_v as 1. Otherwise, output d_v as 0 prompts the storage nodes that the pair (C, H) is invalid.

TrGen $sk \rightarrow T$: a user leverages the secret key to generate a trapdoor and sends it to computation nodes.

- Select a random value $t \in \mathbb{Z}_p$ and compute $T_1 = g^t$. Then, utilize sk to compute $\{T_{2,i} = sk_i^t\}_i^{\sigma}$.
- Generate the trapdoor $T = (T_1, \{T_{2,i}\}_i^{\sigma})$ and send T to computation nodes.

Match $(C, T) \rightarrow d_m$: computation nodes utilize the ciphertext and trapdoor to perform policy verification without compromising user privacy.

- Obtain ciphertexts from the storage nodes. Based on the threshold d, respectively select d values from $\{C_{2,i}\}_i^{\mathcal{P}_r}$ and $\{T_{2,i}\}_i^{\sigma}$ to conduct a set $\mathcal{P}_{r,j}$. Then, for each $\mathcal{P}_{r,j}$, check the equation $\Pi_{i=1}^{\mathcal{P}_{r,j}} e(C_{2,i}, T_{2,i}) \stackrel{?}{=} e(C_3, T_1)$.
- If the equation does not hold for all $\mathcal{P}_{r,j}$, computation nodes output $d_m = 0$. Otherwise, record the set $\mathcal{P}_{r,j}$ and computation nodes output $d_m = 1$. In addition, it means that the user is an authorized reader of this message.
- If d_m equals to 1, computation nodes further select d values from $\{C_{5,i}\}_i^{\mathcal{P}_e}$ and $\{T_{2,i}\}_i^{\sigma}$ to conduct a set $\mathcal{P}_{e,j}$. Then, for each $\mathcal{P}_{e,j}$, check the equation $\Pi_{i=1}^{\mathcal{P}_{e,j}} e(C_{5,i}, T_{2,i}) \stackrel{?}{=} e(C_6, T_1)$. If this equation holds, computation nodes output $d_m = 2$, and it means that the user is an authorized modifier for this message. Then, record the set $\mathcal{P}_{e,j}$.
- If d_m equals to 1, return $C^1 = (\mathcal{P}_{r,j}, \{C_{1,i}\}_i^{\mathcal{P}_r}, C_m)$ to the user. If d_m equals to 2, return $C^2 = (\mathcal{P}_{r,j}, \mathcal{P}_{e,j}, \{C_{1,i}\}_i^{\mathcal{P}_r}, \{C_{4,i}\}_i^{\mathcal{P}_e}, C_m, C_x)$ to the user.

Read $(C^1, sk) \rightarrow m$: receiving the data from the computation nodes, the user utilizes his/her secret key to recover the message.

- Compute $R = \Pi_{i=1}^{\mathcal{P}_{r,j}} e(C_{1,i}, sk_i) = e(g,g)^{\alpha d_1}$. Recover the message $m = C_m \oplus H[R]$.

Edit $(C^2, sk, m^{'}, \mathcal{P}_r^{'}, \mathcal{P}_e^{'}) \rightarrow (C^{'}, H^{'})$: based on the secret key, the user read and edit the message.

- Perform the **Read** algorithm to recover the message m.
- Compute $E = \Pi_{i=1}^{\mathcal{P}_{e,j}} e(C_{4,i}, sk_i) = e(g,g)^{\alpha d_3}$. Recover the message $x = C_x \oplus H[E] \oplus H[R]$. If the user attempts to edit the message, generate a new message $m^{'}$. Subsequently, generate new readability policy $\mathcal{P}_r^{'}$ and editability policy $\mathcal{P}_e^{'}$ and compute the ciphertext $C^{'} = (\{C_{1,i}^{'}\}_i^{\mathcal{P}_r^{'}}, \{C_{2,i}^{'}\}_i^{\mathcal{P}_r^{'}}, C_3^{'}, \{C_{4,i}^{'}\}_i^{\mathcal{P}_e^{'}}, \{C_{5,i}^{'}\}_i^{\mathcal{P}_e^{'}}, C_6^{'}, C_7, C_m^{'}, C_x^{'})$. Next, compute $r_h^{'} = \frac{C_m - C_m^{'}}{x} + r_h$ and generate $H^{'} = (h, r_h^{'})$.
- Send the new pair $(C^{'}, H^{'})$ to computation nodes.

The workflow of PRBFM is shown in Figure 3.

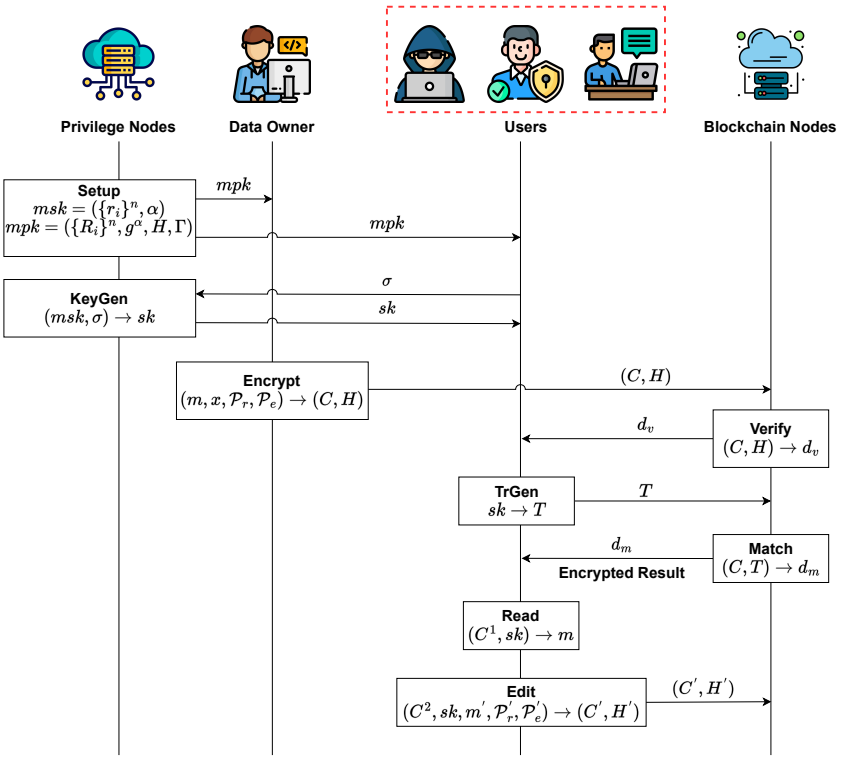

Figure 3. Workflow of PRBFM.

5. Analysis and Discussion

5.1. Correctness Analysis

Theorem 1. *In PRBFM, if and only if an attribute set σ of the user satisfies the policy of the data owner for readers, i.e., $\mathcal{P}_r \subset \sigma$, the $d_m = 1$ and the user is an authorized reader.*

Proof. From the **Encrypt** step, the data owner generates the ciphertext $C = (\{C_{1,i}\}_i^{\mathcal{P}_r}, \{C_{2,i}\}_i^{\mathcal{P}_r}, C_3, \{C_{4,i}\}_i^{\mathcal{P}_e}, \{C_{5,i}\}_i^{\mathcal{P}_e}, C_6, C_7, C_m, C_x)$ and hash value $H = (h, r_h)$. Then, we concentrate on the correctness in the **Match** and the **Read** step. In the **Match** step, after receiving the trapdoor $T = (T_1, \{T_{2,i}\}_i^{\sigma})$, the computation nodes first respectively select d values from $\{C_{2,i}\}_i^{\mathcal{P}_r}$ and $\{T_{2,i}\}_i^{\sigma}$ to conduct a set $\mathcal{P}_{r,j}$ according to the threshold d. Then, for each $\mathcal{P}_{r,j}$, the computation nodes compute

$$\begin{aligned} \Pi_{i=1}^{\mathcal{P}_{r,j}} e(C_{2,i}, T_{2,i}) &= \Pi_{i=1}^{\mathcal{P}_{r,j}} e(R_i^{d_2}, sk_i^t) \\ &= \Pi_{i=1}^{\mathcal{P}_{r,j}} e(g^{d_2 \cdot r_i}, g^{\frac{q(i) \cdot \Delta_{i,\sigma}(0) \cdot t}{r_i}}) \\ &= e(g,g)^{d_2 \cdot t \cdot \sum_{i=1}^{\mathcal{P}_{r,j}} (q(i) \cdot \Delta_{i,\sigma}(0))} \\ &= e(g^{\alpha d_2}, g^t) \\ &= e(C_3, T_1). \end{aligned} \tag{5}$$

Therefore, in the **Match** step, the correctness of PRBFM holds. In the **Read** step, a user gets $C^1 = (\mathcal{P}_{r,j}, \{C_{1,i}\}_i^{\mathcal{P}_r}, C_m)$, and H from the computation nodes. Next, based on his/her secret key sk, the user computes

$$\begin{aligned} & C_m \oplus H[\Pi_{i=1}^{\mathcal{P}_{r,j}} e(C_{1,i}, sk_i)] \\ &= C_m \oplus H[\Pi_{i=1}^{\mathcal{P}_{r,j}} e(g^{d_1 \cdot r_i}, g^{\frac{q(i)\Delta_{i,\sigma}(0)}{r_i}})] \\ &= C_m \oplus H[e(g,g)^{d_1 \cdot \sum_{i=1}^{\mathcal{P}_{r,j}} (q(i) \cdot \Delta_{i,\sigma}(0))}] \\ &= m \oplus H[e(g,g)^{\alpha d_1}] \oplus H[e(g,g)^{\alpha d_1}] \\ &= m. \end{aligned} \tag{6}$$

On the basis of the Lagrange interpolation theorem, if and only if the policy $\mathcal{P}_r$ satisfies $\mathcal{P}_r \subset \sigma$, the message m can be recovered and Equations (5) and (6) hold. Therefore, Theorem 1 is proven. □

Theorem 2. *In PRBFM, if and only if an attribute set σ of the user satisfies the policy of the data owner for modifiers, i.e., $\mathcal{P}_e \subset \sigma$, the $d_m = 2$ and the user is an authorized modifier.*

Proof. Similar to Theorem 1, we first prove the correctness of evaluating whether a given user is an authorized modifier in the **Match** step. Concretely, the computation nodes first select d values from $\{C_{5,i}\}_i^{\mathcal{P}_e}$ and $\{T_{2,i}\}_i^{\sigma}$ to conduct a set $\mathcal{P}_{e,j}$. Then, for each $\mathcal{P}_{e,j}$, the computation nodes compute

$$\begin{aligned} \Pi_{i=1}^{\mathcal{P}_{e,j}} e(C_{5,i}, T_{2,i}) &= \Pi_{i=1}^{\mathcal{P}_{e,j}} e(g^{d_4 \cdot r_i}, g^{\frac{q(i) \cdot \Delta_{i,\sigma}(0) \cdot t}{r_i}}) \\ &= e(g,g)^{d_4 \cdot t \cdot \sum_{i=1}^{\mathcal{P}_{e,j}} (q(i) \cdot \Delta_{i,\sigma}(0))} \\ &= e(C_6, T_1). \end{aligned} \tag{7}$$

Then, in the **Edit** step, if and only if $\mathcal{P}_e \subset \sigma$, the user can recover the chameleon secret key x as follows:

$$\begin{aligned} C_x \oplus H[E] \oplus H[R] &= C_x \oplus H[e(g,g)^{d_3 \cdot \sum_{i=1}^{\mathcal{P}_{e,j}} (q(i) \cdot \Delta_{i,\sigma}(0))}] \oplus H[R] \\ &= C_x \oplus H[e(g,g)^{\alpha d_3}] \oplus H[e(g,g)^{\alpha d_1}] \\ &= x. \end{aligned} \tag{8}$$

where $E = \Pi_{i=1}^{\mathcal{P}_{e,j}} e(C_{4,i}, sk_i)$ and $R = \Pi_{i=1}^{\mathcal{P}_{r,j}} e(C_{1,i}, sk_i)$. Next, based on the x recovered according to Equation (8), the user computes $r_h^{'} = \frac{C_m - C_m^{'}}{x} + r_h$. Notably, if and only if $r_h^{'}$ is calculated as mentioned above, the following equation holds:

$$g^{C_m} \cdot C_7^{r_h} \stackrel{?}{=} g^{C_m^{'}} \cdot C_7^{r_h{'}}. \tag{9}$$

Otherwise, the x is not valid, and the modification is recognized as illegal. Therefore, Theorem 2 is proven. □

5.2. Security Analysis

The security properties of PRBFM are formally defined based on its construction. These include privacy security, which encompasses data privacy, policy privacy, and attribute privacy, as well as user collusion resistance and collision resistance. The security model follows the oracle model and is indistinguishable under a chosen-ciphertext attack (IND-CCA). The experiment involves a PPT adversary $\mathcal{A}$ and a challenger $\mathcal{C}$ and is depicted in detail in Figure 4.

Setup: The challenger $\mathcal{C}$ runs the **Setup** step to generate the master secret key $msk = (\{r_i\}^n, \alpha)$ and master public key $mpk = (\{R_i\}^n, g^{\alpha}, H, \Gamma)$. Next, the challenger $\mathcal{C}$ publishes mpk to the adversary $\mathcal{A}$.
Phase 1: The challenger $\mathcal{C}$ permits adversary $\mathcal{A}$ to request the secret keys. In particular, the adversary $\mathcal{A}$ sends the attributes σ to $\mathcal{C}$. The challenger $\mathcal{C}$ randomly generates a $(d-1)$-degree polynomial $q(x)$ and obtains the secret key $sk = \{g^{\frac{q(i)\Delta_{i,\sigma}(0)}{r_i}}\}_i^{\sigma}$.
Challenge: Adversary $\mathcal{A}$ selects the target user and two messages m_0 and m_1, where $|m_0| == |m_1|$. The challenger $\mathcal{C}$ flips a fair binary coin $b \in \{0,1\}$ outside of the view of $\mathcal{A}$. $\mathcal{C}$ encrypts m_0 and sends (C_0, H_0) to the adversary $\mathcal{A}$ if $b = 0$. Otherwise, m_1 is encrypted, and (C_1, H_1) is sent to $\mathcal{A}$. Notably, the secret key of the target user is not able to be requested in ***Phase 1***.
Phase 2: The challenger $\mathcal{C}$ performs similarly as it did in ***Phase 1***, but $\mathcal{A}$ is able to request the secret key of the target user in ***Phase 2***.
Guess: The adversary $\mathcal{A}$ will submit a guess $b^{'}$ of b, and its advantage to win the experiment can be represented as $Adv_{\mathcal{A}[b'=b]}$.

Figure 4. The experiment played between $\mathcal{A}$ and $\mathcal{C}$.

Theorem 3. *For two multiplicative groups $(\mathbb{G}, \mathbb{G}_T, e)$, and a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, if the Decisionla MBDH problem is hard in $(\mathbb{G}, \mathbb{G}_T, e)$, any PPT adversaries are not able to violate the read-only ciphertext indistinguishability by adopting the chosen-ciphertext attack in the random oracle model.*

Proof. We here assume that if simulator $\mathcal{B}$ has the tuple $(A, B, C, Z) = (g^a, g^b, g^c, e(g,g)^z)$, he will manage to distinguish whether $e(g,g)^z$ is equal to $e(g,g)^{\frac{ab}{c}}$ in the simulation. The detailed process of the simulation is illustrated as follows:

Game 0. Game 0 is the original game, where nothing is different from the original scheme.

Game 1. Game 1 differs from Game 0 in the **Setup** step. For any $i \in \sigma$: $R_i = g^{\frac{1}{\xi_i}}$; for $i \notin \sigma$: $R_i = g^{\frac{1}{\omega_i}}$. Next, simulator $\mathcal{B}$ gives the public parameters to adversary $\mathcal{A}$. From adversary $\mathcal{A}$'s view, the received public parameters are indistinguishable from the ones in

Game 0. The adversary $\mathcal{A}$ will terminate the game and return fail if it can tell the difference with the advantage ε_{DL}.

Game 2. Game 2 differs from Game 0 in the **KeyGen** step. Adversary $\mathcal{A}$ requests for the secret key of the user. For any $i \in \sigma$: $sk_i = C^{\zeta_i q(i) \Delta_{i,\sigma}(0)}$; for $i \notin \sigma$: $sk_i = C^{\omega_i q(i) \Delta_{i,\sigma}(0)}$. Next, simulator $\mathcal{B}$ gives the secret key sk to adversary $\mathcal{A}$. The adversary $\mathcal{A}$ will terminate the game and return fail if it can tell the difference with the advantage ε_{DL}.

Game 3. Game 3 differs from Game 2 in terms of the hash oracle $\mathcal{O}_H$. Simulator $\mathcal{B}$ submits the message m to $\mathcal{O}_H$ and receives the requested hash value. During the i-th request, simulator $\mathcal{B}$ generates the hash value $H(i) = msg_i$ and stores $H(i)$ as $|msg_i| = |m|$. The adversary $\mathcal{A}$ will terminate the game and return fail if it can tell the difference with the advantage ε_H.

Game 4. Game 4 differs from Game 3 in terms of **Challenge**. The adversary $\mathcal{A}$ submits two challenge messages m_0 and m_1 to simulator $\mathcal{B}$, where $|m_0| = |m_1|$. The simulator $\mathcal{B}$ flips a coin b and returns a challenge encryption C_b. Then, for $i \in \sigma$, simulator $\mathcal{B}$ computes

$$\begin{aligned} C_{m^*} &= m \oplus H[Z] \\ C^*_{1,i} &= B^{d_1} \\ C^*_{2,i} &= C^{d_2} \\ C^*_3 &= A^{\eta_i}. \end{aligned} \tag{10}$$

For $i \notin \varepsilon$, $\mathcal{B}$ computes

$$\begin{aligned} C_{m^*} &= m \oplus H[Z] \\ C^*_{1,i} &= B^{\frac{\mu_{i,1}}{\omega_i}} \\ C^*_{2,i} &= C^{\frac{\mu_{i,2}}{\omega_i}} \\ C^*_3 &= A^{\eta_i}. \end{aligned} \tag{11}$$

If $Z = e(g,g)^{\frac{ab}{c}}$ and $\alpha^{'} = \frac{b}{c}$, we then obtain $C_{m^*} = m \oplus H[Z] = m \oplus H[e(g,g)^{\frac{ab}{c}}] = m \oplus H[e(g,g)^{a\alpha^{'}}]$, $C^*_3 = A^{\eta_i} = g^{a\eta_i}$. For $i \in \sigma$, $C^*_{1,i} = B^{d_1} = g^{bd_1}$ and $C^*_{2,i} = C^{d_2} = g^{cd_2}$; for $i \notin \sigma$, $C^*_{1,i} = B^{\frac{\mu_{i,1}}{\omega_i}} = g^{b\frac{\mu_{i,1}}{\omega_i}}$ and $C^*_{2,i} = C^{\frac{\mu_{i,2}}{\omega_i}} = g^{c\frac{\mu_{i,2}}{\omega_i}}$.

If $Z = e(g,g)^Z$, because z is randomly selected, Z will be random from the view of the adversary $\mathcal{A}$. The game will be terminated if $\mathcal{A}$ can tell the difference between Game 4 and Game 3. Otherwise, $\mathcal{B}$ is able to solve the MBDH problem. Concretely, if $Z = e(g,g)^z$, $Pr[b^{'} = b | Z = e(g,g)^z] = Pr[b^{'} \neq b | Z = e(g,g)^z] = \frac{1}{2}$. If $Z = e(g,g)^{\frac{ab}{c}}$, the adversary $\mathcal{A}$ has an advantage $\varepsilon^{'}$ to break the game. Next, the possibility of $\mathcal{A}$'s guess $b = b^{'}$ is $Pr[b^{'} = b | Z = e(g,g)^{\frac{ab}{c}}] = \frac{1}{2} + \varepsilon^{'}$. Therefore, the overall advantage of the simulator $\mathcal{B}$ for breaking the MBDH game is $Adv^{MBDH}_{\mathcal{B}}[b^{'} = b] = |\frac{1}{2} Pr[b^{'} = b | Z = e(g,g)^z] + \frac{1}{2} Pr[b^{'} = b | Z = e(g,g)^{\frac{ab}{c}}] - \frac{1}{2}| = \frac{1}{2}\varepsilon^{'}$.

Game 5. The adversary $\mathcal{A}$ and the simulator $\mathcal{B}$ play the game as mentioned before, except $\sigma \neq \sigma^*$. If adversary $\mathcal{A}$ can tell the difference between **Game 4** and **Game 5**, simulator $\mathcal{B}$ has the advantage ε_{DL} to break the discrete logarithm (DL) problem. Therefore, we obtain $Adv_{\mathcal{A}}[b^{'} = b] = 2\varepsilon_{DL} + \epsilon_H + \frac{1}{2}\varepsilon^{'}$ as the advantage for the adversary $\mathcal{A}$ to win the game. Because the MBDH problem is hard, the components in $Adv_{\mathcal{A}}[b^{'} = b]$ are all negligible. Hence, the $Adv_{\mathcal{A}}[b^{'} = b]$ is also negligible. Next, (C^*_0, H^*_0) and (C^*_1, H^*_1) are indistinguishable from the view of the adversary $\mathcal{A}$. Hence, the PRBFM scheme is able to defend against the chosen-ciphertext attack, and Theorem 3 is proven. □

Theorem 4. *The security of PRBFM includes data privacy, attribute privacy, and policy privacy.*

Proof. Obviously, the security of PRBFM includes data privacy. When it comes to attribute privacy, because the communication channel is secure, the user's attribute set σ can be pre-

vented from being exposed to other entities except for selected privilege nodes during the **KeyGen** step. Next, in the **Match** step, the attribute privacy means the indistinguishability of $T_0 = (T_{0,1}, \{T_{0,2,i}\}_i^{\sigma_0})$ and $T_1 = (T_{1,1}, \{T_{1,2,i}\}_i^{\sigma_1})$. In particular, the simulator $\mathcal{B}$ first selects a random value $\alpha \in \mathbf{Z}_p$. Then, it computes $T_b = (T_{b,0} = g^{\alpha}, \{T_{b,1,i} = g^{sk_i^{\alpha}}\}^{\sigma_b})$ and sends T_b to adversary $\mathcal{A}$. Next, the adversary $\mathcal{A}$ has a negligible advantage ε to guess whether $b = b^{'}$ as follows:

$$Adv_{\mathcal{A}}^{Attr-Pri}[b^{'} = b|\mathcal{A}(\sigma_0, \sigma_1, T_b)] \leq \varepsilon. \tag{12}$$

Hence, adversary $\mathcal{A}$ is not able to violate the attribute privacy during the **Match** step. For other steps such as **Setup**, **Encrypt**, **Verify**, **Read**, and **Edit**, the adversary $\mathcal{A}$ is not able to get information related to the secret key *sk*. Therefore, attribute privacy in PRBFM is guaranteed.

For policy privacy, adversary $\mathcal{A}$ can violate it in two ways. The first is obtaining policies $(\mathcal{P}_r, \mathcal{P}_e)$ from the ciphertext $\mathcal{C}$ generated by the data owner. If the adversary $\mathcal{A}$ is able to obtain $(\mathcal{P}_r, \mathcal{P}_e)$ from $\mathcal{C}$, similar to Theorem 3, simulator $\mathcal{B}$ has the ability to perform a PPT algorithm to break the MBDH problem. However, it is evident that the MBDH problem is hard. Thus, adversary $\mathcal{A}$ is not able to violate the policy privacy from the ciphertext $\mathcal{C}$. □

Theorem 5. *If the correctness of the Lagrange interpolation theorem-based secret sharing scheme [32] holds and Theorem 3 is proven, PRBFM can defend against the user collusion attack.*

Proof. Here, we assume that the user may colludes with other users, but there is no collusion with the blockchain nodes. In the **KeyGen** step, the privilege node randomly generates a Lagrange polynomial $q(x)$. Then, it calculates $q(i)$ and the Lagrange parameters $\{\Delta_{i,\sigma}(0)\}_i^{\sigma}$ for each attribute $i \in \sigma$. In PRBFM, only the privilege nodes can obtain $\{q(i)\}_i^{\sigma}$ and $\{\Delta_{i,\sigma}\}_i^{\sigma}$. Suppose an unauthorized user with $q^{'}(1)\Delta_{1,\sigma}^{'}(0)$ collude with $d-1$ unauthorized users who have $\{q^{'}(i)\Delta_{i,\sigma}^{'}(0)\}_{i=1}^{d}$, and their aim is to violate the readability governance in PRBFM. According to Equation (6), $m = C_m \oplus H[\Pi_{i=1}^{\mathcal{P}_{r,j}} e(C_{1,i}, sk_i)]$. Next, the unauthorized user computes

$$\begin{aligned} & C_m \oplus H[\Pi_{i=1}^{\mathcal{P}_{r,j}} e(C_{1,i}, sk_i)] \\ & = C_m \oplus H[\Pi_{i=1}^{\mathcal{P}_{r,j}} e(g^{d_1 \cdot r_i}, g^{\frac{q^{'}(i)\Delta_{i,\sigma}^{'}(0)}{r_i}})] \\ & = C_m \oplus H[e(g,g)^{d_1 \cdot \sum_{i=1}^{\mathcal{P}_{r,j}} (q^{'}(i) \cdot \Delta_{i,\sigma}^{'}(0))}]. \end{aligned} \tag{13}$$

On the basis of the correctness of the Lagrange interpolation theorem-based secret sharing scheme, we can indicate that $\sum_{i=1}^{\mathcal{P}_{r,j}} (q^{'}(i) \cdot \Delta_{i,\sigma}^{'}(0))$ is not equivalent to the secret value α. Hence, the user cannot violate the readability governance. The user collusion resistance of the editability governance can also be proved in a similar way as the proof of the readability analyzed above. □

Theorem 6. *If the chameleon hash [16] satisfies the collision resistance, PRBFM can defend against hash collision in the random oracle model.*

Proof. In PRBFM, the Lagrange interpolation theorem is utilized to enhance the chameleon hash-based redactable blockchain with PRBFM and bilateral access control. Thus, the collision resistance problem of PRBFM is equivalent to the collision resistance of the chameleon hash. Hence, Theorem 6 is proven. □

5.3. Complexity Analysis

In Table 1, we present the comparison among some recently proposed schemes and PRBFM from the aspects of computational complexity and space complexity, where l is the size of the identity space in Xu et al.'s [21] scheme.

Table 1. Comparison table of some recently proposed schemes.

Scheme	Computational Complexity				Space Complexity			
	KeyGen	**Encrypt**	**Verify**	**Edit**	**System Parameter**	**Encryption Key**	**Decryption Key**	**Ciphertext**
Derler et al.'s [6] scheme	$\mathcal{O}(2\sigma)$	$\mathcal{O}(\sigma^2)$	$\mathcal{O}(1)$	$\mathcal{O}(\sigma^2+2\sigma)$	$\mathcal{O}(1)$	$\mathcal{O}(\sigma)$	$\mathcal{O}(2\sigma)$	$\mathcal{O}(3\sigma)$
Ma et al.'s [7] scheme	$\mathcal{O}(\sigma)$	$\mathcal{O}(2\sigma)$	$\mathcal{O}(1)$	$\mathcal{O}(6\sigma)$	$\mathcal{O}(1)$	$\mathcal{O}(\sigma)$	$\mathcal{O}(2\sigma)$	$\mathcal{O}(4\sigma)$
Xu et al.'s [21] scheme	$\mathcal{O}(\sigma+l)$	$\mathcal{O}(\sigma^2)$	$\mathcal{O}(1)$	$\mathcal{O}(l^2+3l)$	$\mathcal{O}(1)$	$\mathcal{O}(\sigma)$	$\mathcal{O}(\sigma)$	$\mathcal{O}(2\sigma)$
PRBFM	$\mathcal{O}(\sigma)$	$\mathcal{O}(\mathcal{P}_r+\mathcal{P}_e)$	$\mathcal{O}(1)$	$\mathcal{O}(3\sigma)$	$\mathcal{O}(1)$	$\mathcal{O}(\mathcal{P}_r+\mathcal{P}_e)$	$\mathcal{O}(\sigma)$	$\mathcal{O}(\mathcal{P}_r+\mathcal{P}_e)$

For computational complexity, PRBFM is comparable to that of Xu et al.'s and Ma et al.'s schemes and significantly outperforms that of Derler et al.'s scheme. PRBFM's key generation step has a computational complexity of $\mathcal{O}(\sigma)$ because its privilege nodes randomly select one Lagrange polynomial and perform operations. The computational complexity of the encryption algorithm is about the number of attributes and the size of the policy rather than their multiplication. For the verification steps, it only calculates one equation instead of processing more verification steps in Ma et al.'s scheme and Derler et al.'s scheme, which makes its complexity $\mathcal{O}(1)$. When it comes to the **Read** and **Edit** step, the computational complexity of PRBFM is also comparable to other schemes as it is only related to the size of the policy and the number of attributes.

The space complexity of PRBFM's system parameter is fixed because of the utilization of collision-resistant hash functions, making it comparable with other schemes. The space complexity of the **Encrypt**, **Read**, and **Edit** processes are also influenced by the policy size and number of attributes, respectively. Additionally, since they are determined by the size of the policy and the number of attributes, the space complexity of the ciphertext and the trapdoor is comparable to others.

5.4. Application Discussion

This part explores the potential applications of PRBFM in real-world scenarios. As previously demonstrated, PRBFM successfully addresses limitations in terms of security and functionality. Consequently, PRBFM can be applied in diverse and intricate real-world scenarios while ensuring data privacy preservation.

- **Smart Medical:** Drug testing in the smart medical scenario, based on mobile crowdsourcing, can utilize various sensors. Typically, access to these medical data is limited to patients with specific symptoms. However, attaining a 100% match using patients' physiological data is not feasible, considering the variability of these numerical values. Furthermore, since these physiological data belong to the patients themselves, it is crucial to protect their privacy. Consequently, based on its characters, PRBFM can be applied in this scenario to overcome these challenges.
- **Smart Transportation:** Some companies may employ vehicles equipped with sensors to collect and update transportation data for the purpose of offering predictive services. However, to alleviate the server load, only vehicles in specific conditions (e.g., traffic jams) would be granted access to these prediction data, as using the strict match rule to judge the satisfaction of the conditions is unrealistic. Thus, in this scenario, there is a significant need for fuzzy matching with data privacy preservation, making it conducive to adopting PRBFM.

6. Experimental Evaluation

6.1. Experimental Settings

6.1.1. Setup

We have implemented a prototype system to evaluate the experimental performance of PRBFM. The system programming was carried out in Java, utilizing the Java Pairing-Based Cryptography (JPBC) library. Then, the Type A curve with 80-bit security was selected as the symmetric pairing (Type-I). For the blockchain system, we opted for the FISCO blockchain because of its reliability. To test our schemes, we employed five cloud servers with eight CPUs and 32 GB RAM as blockchain nodes. Of these nodes, two were chosen to be the privilege nodes. To simulate users and data owners, volunteers utilized PCs with 16 GB RAM to communicate with the blockchain nodes. We assumed each user would select a maximum of 100 attributes (i.e., affiliation, occupation, and gender) without loss of generality. Each experiment was conducted ten times, and the average cost was recorded as the final result.

6.1.2. Dataset

To show the efficacy of our proposed schemes, we deployed the MHN dataset, which holds millions of published news headlines sourced from the Australian Broadcasting Corporation. In the course of the experiments, the data owner chooses a headline randomly and uploads it to the blockchain nodes.

6.1.3. Baselines for Comparison

We conducted an assessment of the effectiveness of our proposed PRBFM model by comparing it with other recent fine-grained redactable blockchain schemes, including Derler et al.'s scheme [6], Ma et al.'s scheme [7], and Xu et al.'s scheme [21].

6.1.4. Metrics

To more fully evaluate the performance of PRBFM, we will measure it by four types of metrics:

- The running time for key generation;
- Time consumption on the data owner side;
- Time consumption on the user side;
- Consumption on the blockchain node side.

6.2. Experimental Results

First, we measure the performance of the **KeyGen** step on the privilege node regarding computational costs. We varied the number of attributes from 10 to 100, and the results are displayed in Figure 5. The experimental results indicate that the PRBFM schemes are more efficient in comparison to other schemes. For instance, when the size of attributes is set to 100, PRBFM takes only 1 s to complete key generation, while Derler et al.'s scheme, Ma et al.'s scheme, and Xu et al.'s scheme require 17s, 6s, and 10s, respectively. The reason for this disparity is that the privilege nodes in PRBFM only select one Lagrange polynomial randomly and perform operations with $\mathcal{O}(\sigma)$ complexity. In contrast, the three other proposed schemes require pairing operations or hash calculations to obtain the secret key.

To measure the time consumption on the data owner side, we set the policy range from 10 to 100, as shown in Figure 6. Derler et al.'s scheme and Xu et al.'s scheme incur larger time consumption due to extra exponent operations. In contrast, Ma et al.'s scheme and PRBFM exhibit better performance. Additionally, the Adapt step in Ma et al.'s scheme requires extra pairing operations for policy matching, resulting in larger time consumption for Ma et al.'s scheme than PRBFM.

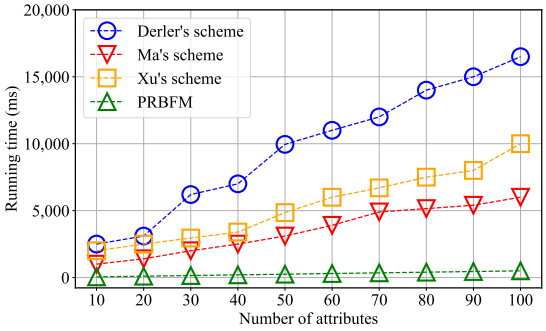

Figure 5. Running time for key generation.

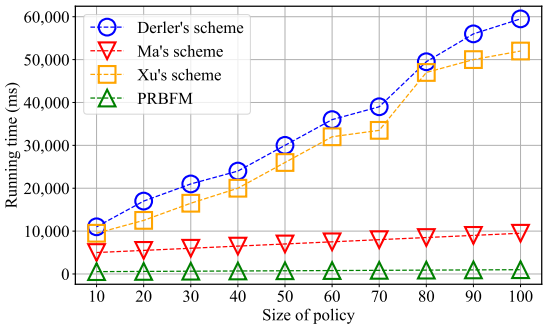

Figure 6. Time consumption on the data owner side.

Regarding Figure 7a, the time consumption is evaluated with the policy size ranging from 10 to 100. The practicality of the **Verify** step in all schemes for real-world scenarios is evident. Notably, PRBFM only needs to verify the generic trapdoor, while Derler et al.'s scheme and Ma et al.'s scheme are required to verify both the generic and ephemeral trapdoor. Therefore, the time consumption for Derler et al.'s and Ma et al.'s schemes is comparatively higher than PRBFM's. Additionally, Xu et al.'s scheme requires more time-consuming operations during the verification step, resulting in it having a higher time consumption than any other scheme.

We then evaluate the time consumption of the trapdoor generation in terms of the user and blockchain nodes. More specifically, we randomly select multiple policy sets with varying data sizes. We denote the total number of policy sets as A_p. Initially, the user is required to obtain the trapdoor related to their attributes, with the number of requests ranging from 10 to 50. Figure 7b indicates that when a user sends 50 data requests to PCBR with $A_p = 5$, the time consumption is 26 s, which is considered practical for real-world scenarios.

During the **Match** stage, we experimented with varying the size of the policy between 10 and 100. To illustrate the results of the experiment for the readability and editability policy match, Figure 8a,b, respectively, display the outcomes. We observed that as the size of the policy increased, the time taken to complete the process increased as well for both readability and editability policy match. Moreover, both matching processes took less than 2 s, indicating the viability of PRBFM. Notably, implementing some cryptographic techniques, such as those presented in [33,34], may improve the efficiency of our proposed strategies. Due to space constraints, we reserve the investigation of streamlining our techniques for future research.

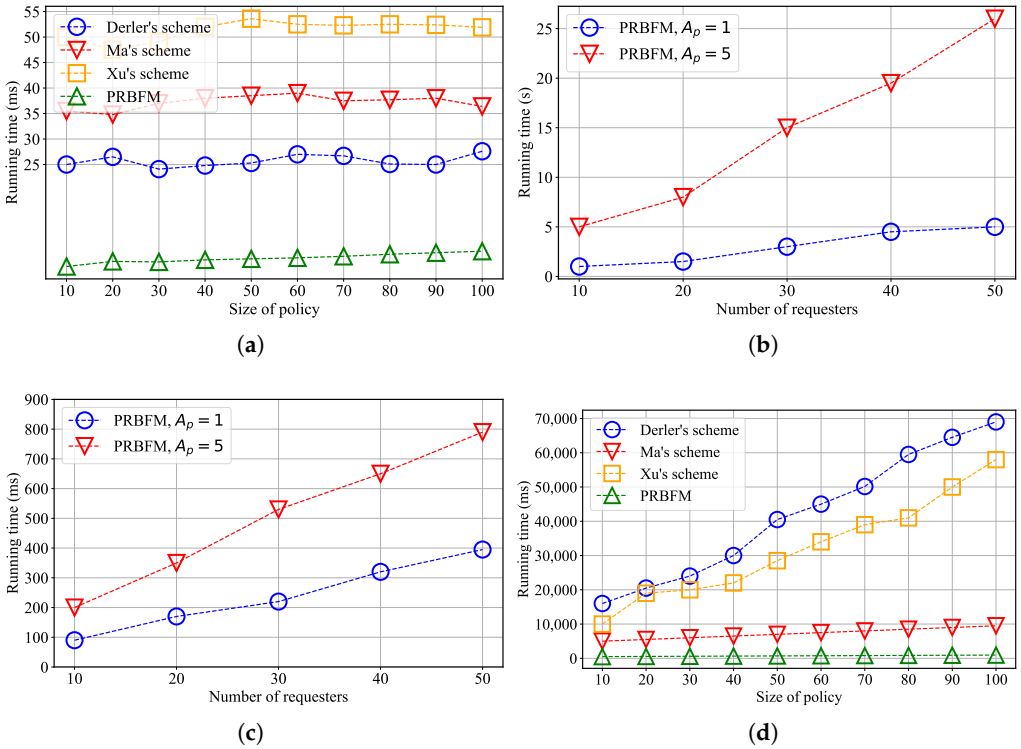

Figure 7. Time consumption on the user side. (**a**) Running time for verification; (**b**) Running time for trapdoor generation; (**c**) Running time for authorized readers; (**d**) Running time for authorized modifiers.

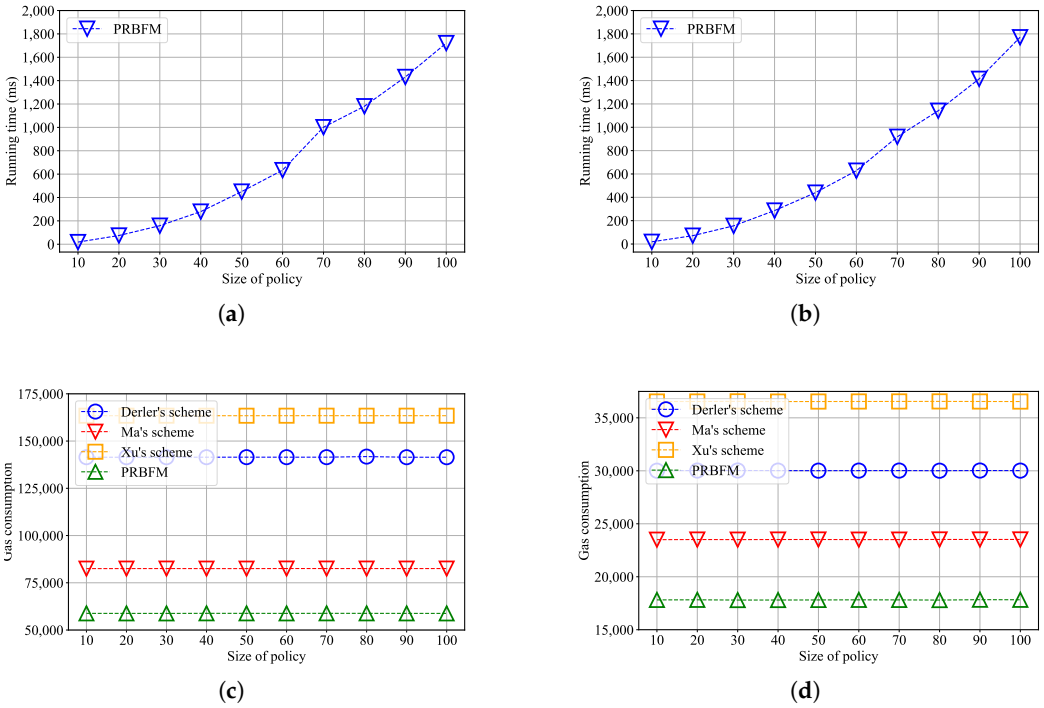

Figure 8. Consumption on blockchain node side. (**a**) Running time for readability policy matching; (**b**) Running time for editability policy matching; (**c**) Gas consumption for hash upload; (**d**) Gas consumption for hash request.

Next, we evaluate the time consumption for the **Read** step in terms of authorized readers. In Figure 7c, the number of requesters ranges from 10 to 50, while the value of

A_p is set at 1 and 5. Since in the PRBFM scheme, the set $\mathcal{P}r, j$ is utilized by authorized readers to recover the message with linear complexity while delegating the policy-matching process to the blockchain nodes, the time consumption for message reading is relatively low. As illustrated in Figure 7c, even with five separate policy sets ($Ap = 5$) and 50 requests, an authorized reader can read messages within 0.8 s.

In this part, we assessed the time consumption among Derler et al.'s scheme, Ma et al.'s scheme, Xu et al.'s scheme, and PRBFM for authorized modifiers. As demonstrated in Figure 7d, PRBFM is more effective than the other schemes when it comes to message editing. This outcome is due to the better performance of PRBFM in the two significant operations (i.e., chameleon secret key recovery and the **Encrypt** step) of the **Edit** step.

Finally, we evaluated the proposed schemes' performance in terms of costs (i.e., gas) on the FISCO blockchain. We varied the policy size from 10 to 100, and the experimental results are exhibited in Figure 8c,d. As illustrated, the gas cost for PRBFM is lower than that of the other schemes. This result is due to PRBFM exclusively updating or requesting one hash value, whereas Ma et al.'s scheme, Derler et al.'s scheme, and Xu et al.'s scheme involve 2, 4, and 6 values, respectively.

7. Related Works

In this section, we introduce the existing studies of the redactable blockchain.

Ateniese et al. [16] proposed the idea of a redactable blockchain. Their scheme uses a permissioned blockchain that needs a central authority to grant rewriting privileges to a designated party, referred to as the "modifier." The designated modifier could coordinate and delete specific content from the blockchain by utilizing a large-scale MPC protocol. This protocol enables block-level rewriting with coarse-grained control, achieved through the use of public key infrastructure (PKI), and an enhanced collision-resistant chameleon hash [31]. The enhanced collision-resistant chameleon hash facilitates modification of the hash link between block headers, while the PKI ensures the sealing of the chameleon hash trapdoor.

Derler et al. [6] introduced a permissioned redactable blockchain that allows for flexible rewriting of privileges through transaction-level rewriting controlled at a fine-grained level. Unlike the research conducted in [16], the authors of [6] employed chameleon hash to hash the transactions during the computation of the Merkle root. To formalize their solution, they presented a new concept called policy-based chameleon hash (PCH), which is derived from chameleon hashes with ephemeral trapdoors (CHET) [35] and attribute-based encryption (ABE) [36].

Deuber et al. [37] presented a permissionless block-level redactable blockchain that eliminates the requirement for a trusted central authority through the use of consensus-based voting. In their approach, the block header is modified to include two hash links. If a proposal for modification garners enough votes, it is approved by breaking one of the links while leaving the other intact. However, this scheme has the drawback of providing weak accountability, as the identity of the modifier can be traced from the proposed pool. Furthermore, this redactable blockchain inherits the vulnerability of consensus-based voting algorithms, rendering it susceptible to bribing and selfish mining attacks.

Regarding the revocability of the redactable blockchain, Panwar et al. [38] introduced a permissioned redactable blockchain that incorporates dynamic group signature schemes (DGSS) and revocable FAME (RFAME) to achieve traceability and revocability. Traceability ensures the life cycle of all transactions is honestly recorded in the blockchain for assisting in tracking history transactions, while revocability can be used to revoke the editing authority of malicious modifiers. However, traceability offers limited accountability, as accountability is to identify who made a particular transaction or redaction, which is more complicated than traceability. Moreover, the secure assignment of the updated secret key is crucial. To address this challenge, Xu et al. [8] introduced the concept of revocable policy-based chameleon hash (RPCH) and implemented it to facilitate efficient privilege revocation. The cost of implementing revocability in RPCH is almost insignificant compared to FAME [36].

To address the requirement for accountability in a redactable blockchain, Tian et al. [18] were the first to explore accountability using a policy-based chameleon hash (PCH). However, their solution only supports weak accountability. Specifically, it could only link the modified transaction while lacking the ability to identify potential key leakage. To address these issues, Xu et al. [21] presented a novel design of PCH called black-box accountability (PCHA) and introduced a practical attribute-based traitor tracing (ABTT) scheme with adaptive security.

For rewriting authorization in a decentralized environment, Zhang et al. [17] presented a multi-authority policy-based chameleon hash (MPCH), and Ma et al. [7] introduced a decentralized policy-based chameleon hash (DPCH). Both MPCH and DPCH employ CHET for data rewriting management and utilize multi-authority attribute-based encryption (MA-ABE) to handle rewriting privileges. However, their schemes do not support dynamic nodes, as the departure of participants would result in a single point of failure. Therefore, Zhang et al. [22] propose a novel dynamic and decentralized attribute-based chameleon hash (DACH) to enable the blockchain history's mutability. By leveraging DACH, their scheme achieves a decentralized, secure, and dynamically redactable blockchain (SDRchain).

Xu et al. [39] introduced a novel redactable blockchain named k-time modifiable and epoch-based redactable blockchain (KERB), which draws inspiration from the double-authentication preventing signature (DAPS) to regulate rewriting privileges. However, due to the restrictions on the number of k-times and epochs, it is necessary to combine customized k-times and epochs, which has yet to be accomplished. To tackle this challenge, Liu et al. [24] selected times and epochs as the controlling factors, restricting users from invoking the credential show method based on customized times within each epoch established by the certificate authority. In their approach, users can redact their credentials to achieve selective disclosure.

Shen et al. [23] propose a verifiable and redactable blockchain that allows for full editing operations, thus supporting verifiability. Their scheme successfully combines the complete editability of block objects and the verifiability of the blockchain state with reasonable additional costs. Specifically, the authors have built a redactable blockchain using a double trapdoor chameleon hash family, which allows for computationally efficient block editing while maintaining resistance against key exposure. However, these presented schemes ignore the privacy of sensitive data and do not support error tolerance during policy matching. Thus, the PRBFM scheme proposed in this paper is designed for fine-grained redactable blockchain with policy fuzzy matching in a privacy-preserving manner.

8. Conclusions

In this paper, we proposed a novel privacy-preserving fine-grained redactable blockchain with fuzzy policy matching for mobile crowdsensing scenarios. Firstly, we presented the PRBFM scheme utilizing Lagrange secret sharing and chameleon hash. Then, we combined the policy paradigm in PRBFM to further design a privacy-preserving policy matching delegation mechanism to protect policy privacy and reduce user overhead. Additionally, we formally analyzed the security of our scheme under IND-CCA, and also provided a detailed complexity analysis in terms of computational and special. Finally, we implemented and measured our scheme on the FISCO blockchain and demonstrated that our scheme outperforms existing works. For future work, we intend to further reduce user overhead with the use of bilateral access control and provide more fine-grained permission management (e.g., usability, readability, and editability).

Author Contributions: Methodology, M.Z.; Software, Y.X.; Formal analysis, T.W.; Investigation, M.Z.; Writing—original draft, H.L.; Supervision, H.G., J.X. and L.Z.; Project administration, H.G. and L.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (Grant Nos. 62202051, 62102027, and 62172042), China Postdoctoral Science Foundation (Grant Nos. 2021M700435 and 2021TQ0042), Guangdong Provincial Key Laboratory of Novel Security Intelligence Technologies

(Grant No. 2022B1212010005), Shandong Provincial Key Research and Development Program (Grant No. 2021CXGC010106), Major Scientific and Technological Innovation Projects of Shandong Province (2020CXGC010116), and Beijing Institute of Technology Research Fund Program for Young Scholars.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zou, S.; Xi, J.; Wang, H.; Xu, G. CrowdBLPS: A blockchain-based location-privacy-preserving mobile crowdsensing system. *IEEE Trans. Ind. Inform.* **2019**, *16*, 4206–4218. [CrossRef]
2. Huang, J.; Kong, L.; Dai, H.N.; Ding, W.; Cheng, L.; Chen, G.; Jin, X.; Zeng, P. Blockchain-based mobile crowd sensing in industrial systems. *IEEE Trans. Ind. Inform.* **2020**, *16*, 6553–6563. [CrossRef]
3. Wang, W.; Yang, Y.; Yin, Z.; Dev, K.; Zhou, X.; Li, X.; Qureshi, N.M.F.; Su, C. BSIF: Blockchain-based secure, interactive, and fair mobile crowdsensing. *IEEE J. Sel. Areas Commun.* **2022**, *40*, 3452–3469. [CrossRef]
4. Zhang, C.; Zhao, M.; Zhu, L.; Zhang, W.; Wu, T.; Ni, J. FRUIT: A blockchain-based efficient and privacy-preserving quality-aware incentive scheme. *IEEE J. Sel. Areas Commun.* **2022**, *40*, 3343–3357.
5. Zhang, C.; Zhao, M.; Zhu, L.; Wu, T.; Liu, X. Enabling efficient and strong privacy-preserving truth discovery in mobile crowdsensing. *IEEE Trans. Inf. Forensics Secur.* **2022**, *17*, 3569–3581. [CrossRef]
6. Derler, D.; Samelin, K.; Slamanig, D.; Striecks, C. Fine-grained and controlled rewriting in blockchains: Chameleon-hashing gone attribute-based. *Cryptol. Eprint Arch.* **2019**.
7. Ma, J.; Xu, S.; Ning, J.; Huang, X.; Deng, R.H. Redactable blockchain in decentralized setting. *IEEE Trans. Inf. Forensics Secur.* **2022**, *17*, 1227–1242. [CrossRef]
8. Xu, S.; Ning, J.; Ma, J.; Xu, G.; Yuan, J.; Deng, R.H. Revocable policy-based chameleon hash. In Proceedings of the Computer Security—ESORICS 2021: 26th European Symposium on Research in Computer Security, Darmstadt, Germany, 4–8 October 2021; Proceedings, Part I 26; Springer: Cham, Switzerland, 2021; pp. 327–347.
9. Jia, M.; Chen, J.; He, K.; Du, R.; Zheng, L.; Lai, M.; Wang, D.; Liu, F. Redactable Blockchain From Decentralized Chameleon Hash Functions. *IEEE Trans. Inf. Forensics Secur.* **2022**, *17*, 2771–2783. [CrossRef]
10. Voigt, P.; Von dem Bussche, A. The eu general data protection regulation (gdpr). In *A Practical Guide*, 1st ed.; Springer International Publishing: Cham, Switzerland, 2017; Volume 10, pp. 10–5555.
11. Alfonsín, J.M.L. Argentina: The right to be forgotten. In *The Right to Be Forgotten: A Comparative Study of the Emergent Right's Evolution and Application in Europe, the Americas, and Asia*; Springer: Cham, Switzerland, 2020; pp. 239–248.
12. Li, C.; Guan, L.; Wu, H.; Cheng, N.; Li, Z.; Shen, X.S. Dynamic Spectrum Control-Assisted Secure and Efficient Transmission Scheme in Heterogeneous Cellular Networks. *Engineering* **2022**, *17*, 220–231. [CrossRef]
13. Zhang, C.; Hu, C.; Wu, T.; Zhu, L.; Liu, X. Achieving Efficient and Privacy-Preserving Neural Network Training and Prediction in Cloud Environments. *IEEE Trans. Dependable Secur. Comput.* 2022, *early access*. [CrossRef]
14. Hu, C.; Zhang, C.; Lei, D.; Wu, T.; Liu, X.; Zhu, L. Achieving Privacy-Preserving and Verifiable Support Vector Machine Training in the Cloud. *IEEE Trans. Inf. Forensics Secur.* **2023**, *18*, 3476–3491. [CrossRef]
15. Li, J.; Ma, H.; Wang, J.; Song, Z.; Xu, W.; Zhang, R. Wolverine: A Scalable and Transaction-Consistent Redactable Permissionless Blockchain. *IEEE Trans. Inf. Forensics Secur.* **2023**, *18*, 1653–1666. [CrossRef]
16. Ateniese, G.; Magri, B.; Venturi, D.; Andrade, E. Redactable blockchain–or–rewriting history in bitcoin and friends. In Proceedings of the 2017 IEEE European Symposium on Security and Privacy (EuroS&P), Paris, France, 26–28 April 2017; IEEE: New York, NY, USA, 2017; pp. 111–126.
17. Zhang, Z.; Li, T.; Wang, Z.; Liu, J. Redactable transactions in consortium blockchain: Controlled by multi-authority CP-ABE. In Proceedings of the Information Security and Privacy: 26th Australasian Conference, ACISP 2021, Virtual Event, 1–3 December 2021; Proceedings 26; Springer: Cham, Switzerland, 2021; pp. 408–429.
18. Tian, Y.; Li, N.; Li, Y.; Szalachowski, P.; Zhou, J. Policy-based chameleon hash for blockchain rewriting with black-box accountability. In Proceedings of the Annual Computer Security Applications Conference, Austin, TX, USA, 7–11 December 2020; pp. 813–828.
19. Liu, Z.; Cao, Z.; Wong, D.S. Blackbox traceable CP-ABE: How to catch people leaking their keys by selling decryption devices on ebay. In Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security, Berlin, Germany, 4–8 November 2013; pp. 475–486.
20. Liu, Z.; Cao, Z.; Wong, D.S. Traceable CP-ABE: How to trace decryption devices found in the wild. *IEEE Trans. Inf. Forensics Secur.* **2014**, *10*, 55–68.
21. Xu, S.; Huang, X.; Yuan, J.; Li, Y.; Deng, R.H. Accountable and Fine-Grained Controllable Rewriting in Blockchains. *IEEE Trans. Inf. Forensics Secur.* **2022**, *18*, 101–116. [CrossRef]
22. Zhang, D.; Le, J.; Lei, X.; Xiang, T.; Liao, X. Secure Redactable Blockchain With Dynamic Support. *IEEE Trans. Dependable Secur. Comput.* 2023, *early access*. [CrossRef]
23. Shen, J.; Chen, X.; Liu, Z.; Susilo, W. Verifiable and Redactable Blockchains with Fully Editing Operations. *IEEE Trans. Inf. Forensics Secur.* **2023**, *18*, 3787–3802. [CrossRef]

24. Liu, Y.; He, D.; Feng, Q.; Luo, M.; Choo, K.K.R. PERCE: A Permissioned Redactable Credentials Scheme for a Period of Membership. *IEEE Trans. Inf. Forensics Secur.* **2023**, *18*, 3132–3142. [CrossRef]
25. Zhang, C.; Zhao, M.; Xu, Y.; Wu, T.; Li, Y.; Zhu, L.; Wang, H. Achieving fuzzy matching data sharing for secure cloud-edge communication. *China Commun.* **2022**, *19*, 257–276. [CrossRef]
26. Yin, X.; Zhu, Y.; Hu, J. Contactless fingerprint recognition based on global minutia topology and loose genetic algorithm. *IEEE Trans. Inf. Forensics Secur.* **2019**, *15*, 28–41. [CrossRef]
27. Boutros, F.; Damer, N.; Kirchbuchner, F.; Kuijper, A. Elasticface: Elastic margin loss for deep face recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 1578–1587.
28. Kumar, R.; Kumar, P.; Tripathi, R.; Gupta, G.P.; Islam, A.N.; Shorfuzzaman, M. Permissioned Blockchain and Deep Learning for Secure and Efficient Data Sharing in Industrial Healthcare Systems. *IEEE Trans. Ind. Inform.* **2022**, *18*, 8065–8073. [CrossRef]
29. Shen, B.; Dong, C.; Minner, S. Combating copycats in the supply chain with permissioned blockchain technology. *Prod. Oper. Manag.* **2022**, *31*, 138–154. [CrossRef]
30. Wu, L.; Lu, W.; Xue, F.; Li, X.; Zhao, R.; Tang, M. Linking permissioned blockchain to Internet of Things (IoT)-BIM platform for off-site production management in modular construction. *Comput. Ind.* **2022**, *135*, 103573. [CrossRef]
31. Hunhevicz, J.; Dounas, T.; Hall, D.M. The promise of blockchain for the construction industry: A governance lens. In *Blockchain for Construction*; Springer: Singapore, 2022; pp. 5–33.
32. Shamir, A. How to Share a Secret. *Commun. ACM* **1979**, *22*, 612–613. [CrossRef]
33. Falzon, F.; Markatou, E.A.; Espiritu, Z.; Tamassia, R. Range Search over Encrypted Multi-Attribute Data. *Proc. VLDB Endow.* **2022**, *16*, 587–600. [CrossRef]
34. Servan-Schreiber, S.; Langowski, S.; Devadas, S. Private approximate nearest neighbor search with sublinear communication. In Proceedings of the 2022 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 22–25 May 2022; IEEE: New York, NY, USA, 2022; pp. 911–929.
35. Camenisch, J.; Derler, D.; Krenn, S.; Pöhls, H.C.; Samelin, K.; Slamanig, D. Chameleon-Hashes with Ephemeral Trapdoors: And Applications to Invisible Sanitizable Signatures. In Proceedings of the Public-Key Cryptography—PKC 2017: 20th IACR International Conference on Practice and Theory in Public-Key Cryptography, Amsterdam, The Netherlands, 28–31 March 2017; Proceedings, Part II 20; Springer: Cham, Switzerland, 2017; pp. 152–182.
36. Agrawal, S.; Chase, M. FAME: Fast attribute-based message encryption. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, Dallas, TX, USA, 30 October–3 November 2017; pp. 665–682.
37. Deuber, D.; Magri, B.; Thyagarajan, S.A.K. Redactable blockchain in the permissionless setting. In Proceedings of the 2019 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 19–23 May 2019; IEEE: New York, NY, USA, 2019; pp. 124–138.
38. Panwar, G.; Vishwanathan, R.; Misra, S. ReTRACe: Revocable and traceable blockchain rewrites using attribute-based cryptosystems. In Proceedings of the 26th ACM Symposium on Access Control Models and Technologies, Virtual, 16–18 June 2021; pp. 103–114.
39. Xu, S.; Ning, J.; Ma, J.; Huang, X.; Deng, R.H. K-time modifiable and epoch-based redactable blockchain. *IEEE Trans. Inf. Forensics Secur.* **2021**, *16*, 4507–4520. [CrossRef]