electronics

MDPI

Article

PIRB: Privacy-Preserving Identity-Based Redactable Blockchains with Accountability

Yuhua Xu [1] and Zihan Li [2,*]

[1] School of Computer Science and Technology, Beijing Institute of Technology, Beijing 100081, China; 1120201967@bit.edu.cn
[2] School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing 100081, China
* Correspondence: zihanl@bit.edu.cn

Abstract: In this paper, we propose a privacy-preserving identity-based redactable blockchain (PIRB), the first identity-based redactable blockchain that supports flexible policies while maintaining accountability. Based on digital identities, PIRB enables a knowledge owner to set one policy for a batch of users while preserving policy privacy. Furthermore, similar to state-of-the-art solutions, PIRB draws inspiration from the proxy re-encryption technique to enforce user accountability. The design of PIRB entails addressing two primary technical challenges: firstly, achieving a flexible policy while upholding policy privacy; secondly, establishing accountability measures. To tackle the former challenge, we propose an enhanced identity-based encryption scheme that integrates polynomial function techniques. To address the latter challenge, a distinct identifier is generated for each user and subsequently concealed within the user's secret key. Specifically, following existing schemes, we present the first scheme PIRB-I to cater to one-way access control scenarios, empowering owners to define access policies for designated editors. Additionally, recognizing the needs on the editor side for owner selection, we enhance PIRB-I through the introduction of matchmaking encryption, thereby supporting bilateral access control in a framework denoted as the second scheme PIRB-II. Notably, PIRB-I and PIRB-II involve a trade-off between computational and communication complexities. Specifically, when contrasted with PIRB-I, PIRB-II facilitates editors in owner selection, thereby mitigating editors' communication overheads at the cost of increased computational overheads during policy generation and matching. Theoretical analysis demonstrates the inherent trade-off complexity and the resilience exhibited by PIRB-I and PIRB-II against chosen-plaintext attacks. Extensive experimentation on the FISCO blockchain shows that, compared with the state-of-the-art works, PIRB-I and PIRB-II achieve 200 times and 100 times computational efficiency improvements and 50 times and 60 times communication efficiency improvements on average, respectively.

Keywords: redactable blockchains; digital identity; privacy preservation; accountability

Citation: Xu, Y.; Li, Z. PIRB: Privacy-Preserving Identity-Based Redactable Blockchains with Accountability. *Electronics* **2023**, *12*, 3754. https://doi.org/10.3390/electronics12183754

Academic Editor: Rameez Asif

Received: 25 August 2023
Revised: 1 September 2023
Accepted: 4 September 2023
Published: 5 September 2023

1. Introduction

Currently, the growing interest in blockchain has led to its increasing identity-based applications, i.e., authentication [1–3], database service [4–7], and health monitoring [8,9]. In these applications, blockchain techniques introduce essential features such as immutability, anonymity, and traceability. To illustrate, consider a scenario in which Alice and Bob engage in the storage, exchange, and management of data through a blockchain-based digital identity scheme on a database [10–12]. Immutability serves to safeguard the security of the submitted digital assets. Simultaneously, anonymity ensures identity privacy for the submitter. Furthermore, traceability effectively thwarts any malicious attempts at data submission. Hence, it is obvious that blockchain techniques exhibit significant potential in identity-based scenarios.

The imperative requirement of redactability in identity-based blockchains has gained prominence in the realm of data security. This requirement harmonizes with the stringent

regulations represented by the General Data Protection Regulation (GDPR) [13,14], aiming to uphold users' right to be forgotten and mitigate the dissemination of malicious data. To elucidate, within the framework of an identity-based blockchain on a knowledge marketplace [15], individuals such as Alice, who are knowledge owners, seek to endow their data with an expiration date, thereby preserving their value and veracity. In addition, in the event of Alice introducing malware into the blockchain, the blockchain administrator, Bob, should possess the capability to revoke Alice's malicious data. Alice also needs the capacity to set flexible access policies for editors while preserving privacy to preclude the misuse of editing privileges. Furthermore, to prevent malicious data erasure or tampering, the scheme must effectively institute mechanisms of accountability. Notably, there is a specific concern that Bob, with edit privileges, may exploit the system by maliciously erasing or altering Alice's data for personal gain. These scenarios underscore the crucial role of redactability and accountability in identity-based blockchains, despite the significant challenges they pose.

To satisfy the requirements of an identity-based redactable blockchain, several schemes have been proposed. Chen et al. [16] presented an identity-based chameleon hash scheme without key exposure. Building upon [16], Zhou et al. [17] introduced an innovative identity-based fine-grained redactable blockchain framework. However, the existing identity-based schemes fail to support one policy for a batch of users, thereby constraining the flexibility of privilege distribution, which results in significant overheads when dealing with a batch of users collectively. Additionally, these schemes lack support for accountability, opening the door to potential malicious behaviors. Consequently, our attention is directed toward attribute-based redactable blockchain schemes that facilitate the implementation of flexible policies. Derler et al. [18] initially introduced the policy-based chameleon hash (PCH) approach, harnessing chameleon hash with ephemeral trapdoors (CHET) and attribute-based encryption (ABE). Seeking to bolster accountability, Xu et al. [19] introduced the identity-based signature with existential unforgeability. The access policies in these schemes achieve high flexibility. Regrettably, none of the existing attribute-based schemes have provisions for the preservation of policy privacy [20], potentially leading to privacy breaches when applying existing attribute-based methodologies directly to identity-based scenarios. Specifically, treating each identity as an attribute, during the edit phases of attribute-based redactable blockchain schemes, the policy matrices must be utilized in plaintext. This not only exposes the policy context and size, thus compromising privacy, but also divulges the identities of suitable users during the match process, which can further cause secret key forgery and information leakage, jeopardizing public trust in the blockchain-based digital identity platform. As an illustrative example, within an identity-based blockchain system designed for medical services, Alice, a patient, stores her medical records on the blockchain. She sets a policy that grants the editing privilege to Bob, the designated doctor responsible for ensuring that the records align with Alice's current health condition. However, a potential vulnerability arises with policy disclosure, wherein Bob's identity becomes susceptible to exposure following an editing action. Specifically, if an attacker gains access to the policy's content, they can extract the list of authorized editors' identities and verify that Bob's identity is among them, which could potentially facilitate fraudulent activities and pose threats to the identity-based system's security. Furthermore, while both the existing identity-based and attribute-based schemes concentrate on one-way access control scenarios, none of them satisfies the need to select suitable owners on the editor side to reduce communication costs. This limitation impedes the broader application of redactable blockchain techniques in bilateral access control scenarios, which has become a practical requirement [21–23]. Hence, as shown in Table 1, the research gap pertains to the proposition of an identity-based redactable blockchain with the support of flexible policies while preserving policy privacy, achieving accountability, and supporting bilateral access control.

Table 1. Comparison with redactable blockchain schemes.

Scheme	Paradigm	Policy Privacy	Accountability	Flexible Policy	Access Control
[24]	public-key-based	✘	✘	✘	one-way
[18]	attribute-based	✘	✘	✔	one-way
[25]	consensus-based	✘	✔	✔	one-way
[26]	attribute-based	✘	✔	✔	one-way
[27]	attribute-based	✘	✔	✔	one-way
[28]	attribute-based	✘	✔	✔	one-way
[29]	attribute-based	✘	✘	✔	one-way
[19]	attribute-based	✘	✔	✔	one-way
[16]	identity-based	✔	✘	✘	one-way
[30]	identity-based	✔	✘	✘	one-way
[31]	identity-based	✔	✘	✘	one-way
[17]	identity-based	✔	✘	✘	one-way
PIRB-I	identity-based	✔	✔	✔	one-way
PIRB-II	identity-based	✔	✔	✔	bilateral

The symbols ✔ and ✘ in Table 1 represent "support" and "not support", respectively.

The realization of a practical identity-based redactable blockchain presents three distinct challenges that need to be addressed. Firstly, integrating redactable blockchain functionality within identity-based scenarios requires the formulation of a scheme that supports flexible access policies while upholding policy privacy. Precisely, the scheme should facilitate one policy for a batch of users. Secondly, to enhance the scheme's integrity, accountability measures must be introduced, thereby enabling the imposition of penalties upon users found engaging in malicious behaviors. Thirdly, to meet the imperative of minimizing editors' communication overhead, it becomes crucial to endow editors with the capability to set policies for owner selection. Consequently, the realization of bilateral access control presents itself as a challenge.

In response to the prevailing challenges, we propose a privacy-preserving identity-based redactable blockchain scheme with accountability. Specifically, we summarize our contributions as follows.

- To tackle the challenges, we present a privacy-preserving identity-based redactable blockchain based on the chameleon hash with ephemeral trapdoors, denoted as PIRB, which contains two schemes, PIRB-I and PIRB-II. With an identity-based encryption scheme introduced, PIRB-I facilitates one-way access control, while PIRB-II achieves bilateral access control while upholding match privacy preservation. Moreover, we leverage the polynomial function technique to support one policy for a batch of users.
- To mitigate the potential misuse of editing privileges for malicious behaviors, the proxy re-encryption technique is introduced as an accountability mechanism.
- A formal theoretical analysis establishes the correctness, security, and complexity of PIRB. Correctness analysis proves the necessary and sufficient conditions for the accurate operation of PIRB, contingent upon the fulfillment of the policy by the identity-based attribute. Security analysis demonstrates that PIRB concurrently upholds trapdoor privacy, identity privacy, policy privacy, and match privacy under the chosen-plaintext attack model. Additionally, complexity analysis presents the computational and communication complexity of both PIRB-I and PIRB-II, comparing them with two existing attribute-based redactable blockchain schemes. Empirical experiments corroborate our scheme's practical efficiency enhancements.

Organization. In this paper, we present a comprehensive overview of our work, organized as follows. The problem formulation, which includes the system model, threat model, problem statement, and design goals, is introduced in Section 2. Next, we introduce the preliminaries, which present the definition of the chameleon hash with ephemeral trapdoors and identity-based encryption, in Section 3. Subsequently, Section 4 is dedicated to the detailed construction of PIRB-I and PIRB-II. In Section 5, a formal theoretical analysis, which thoroughly analyzes the correctness, security, and complexity of PIRB-I and PIRB-II, is provided. In Section 6, we present a thorough performance evaluation. The related works are introduced in Section 7. Finally, we give the conclusions of this paper in Section 8.

2. Problem Formulation

2.1. System Model

Within our system, the central emphasis lies on the consortium blockchain, a framework commonly employed for data sharing. The operation of our system is conducted under the oversight of multiple authoritative nodes, ensuring privacy preservation. As depicted in Figure 1, the system model of PIRB encompasses four primary entities: consortium nodes, knowledge owners, knowledge editors, and service nodes. These entities are elaborated upon as follows.

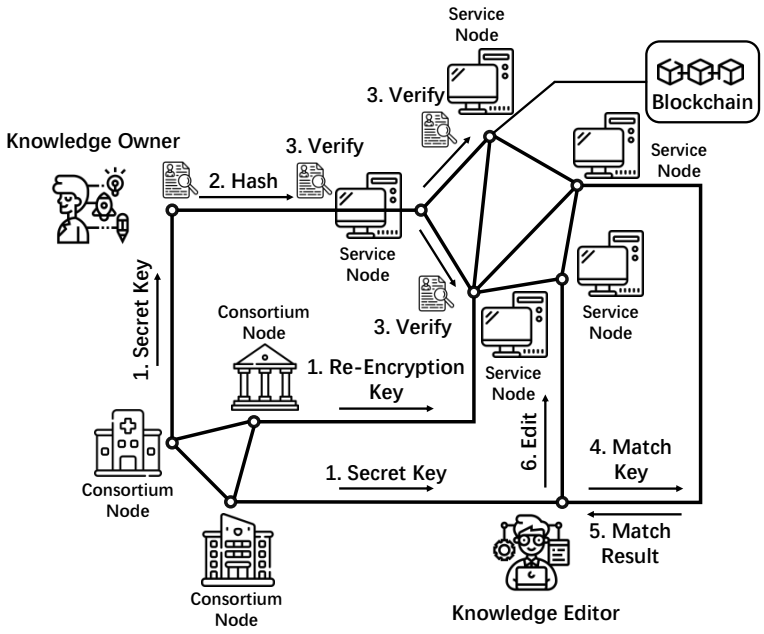

Figure 1. System model of PIRB.

Consortium Node. The responsibility for key generation and the allocation of virtual identities rests with the consortium nodes. These nodes also uphold accountability by managing the association between virtual and actual identities, disclosing such information publicly when required. Typically, these consortium nodes are endowed with authorization due to their representation of prominent enterprises, thereby considered to be fully trusted.

Knowledge Owner. Knowledge owners encompass both entities and individuals who submit data into the blockchain. These proprietors have the authority to grant specific users the edit privilege through the formulation of an identity-based access policy concurrent with the generation of the data hash.

Knowledge Editor. The task of a knowledge editor involves proficiently matching authorized data for editing with corresponding trapdoors of CHET, utilizing service nodes. Subsequently, in finalizing the edit, the knowledge editor calculates a valid message for the edited data, ensuring that all checks hold while maintaining the original data hash.

Service Node. Service nodes are responsible for storing and updating the blockchain, as well as overseeing proxy re-encryption and matching services for knowledge editors.

Specifically, these nodes decentralize the storage of the blockchain and, when engaging in edits, they ensure data consistency through a consensus protocol. Importantly, they also preserve the integrity of the original records for accountability.

2.2. Threat Model

The consortium nodes are bestowed with full trust, engaging in interactions solely through secure channels. Functioning as data submitters, the knowledge owners are also fully trusted. The service nodes are honest-but-curious. This implies that these service nodes faithfully execute computations and adhere to protocols yet concurrently harbor an intent to gather privacy-related information from other entities. Specifically, while matching appropriate editors to messages, the service nodes endeavor to attain the ability to modify messages, thereby attempting to procure unwarranted privileges. Additionally, the service nodes strive to ascertain the identities of users along with deducing interrelationships among these users via the outcomes of matches. Notably, the knowledge editors are positioned as untrusted entities. To elaborate, a knowledge editor endeavors to gain editing rights for messages for which the owners have not been granted. Furthermore, even an editor who possesses edit privileges may exhibit malicious behaviors, seeking to evade subsequent repercussions. It is imperative to acknowledge that the service nodes are devoid of collusion capabilities with other entities, as well as any inclination to feign legitimacy as valid knowledge owners or editors.

2.3. Problem Statement and Design Goals

Certain regulations necessitate the redactability of blockchains for effective management. Nonetheless, the inherent structure of hash links imposes substantial constraints on the redactability, striking a balance between security and flexibility. To address this challenge, some solutions have been proposed in the form of attribute-based redactable blockchain schemes. However, within an identity-based framework, the absence of a redactable blockchain scheme remains conspicuous. Moreover, a straightforward adaptation of existing attribute-based schemes is deemed infeasible due to the potential privacy vulnerabilities in identity-based scenarios. Specifically, considering each identity as an attribute, during edit phases in attribute-based redactable blockchain schemes, the utilization of policies for plaintext purposes lacks the preservation of the policy context and size privacy, consequently enabling the potential inference of the suitable editors' identities. Thus, the main challenge lies in devising a privacy-preserving identity-based redactable blockchain solution that encompasses trapdoor privacy, identity privacy, policy privacy, and match privacy, all while upholding accountability. We delineate a set of design goals as follows.

Privacy. The preservation of privacy in PIRB is of paramount importance, encompassing trapdoor privacy, identity privacy, policy privacy, and match privacy. Specifically, regarding trapdoor privacy, PIRB ensures that an editor's access to the trapdoor of CHET from the ciphertext is contingent upon the satisfaction of the owner's requirements by the editor's identity and policy. Concerning identity privacy, PIRB ensures that only the consortium nodes possess the capability to differentiate the actual identity of a participating user from that of their peers. Both users and service nodes remain incapable of distinguishing, in the absence of consortium nodes, the submitting owner or editor from other entities. Turning to policy privacy, PIRB prevents any derivation of the user's policy content or size from the ciphertext. With only a user's ciphertext, discerning a fitting identity for the policy or ascertaining the precise policy dimension becomes an infeasible task. Lastly, match privacy necessitates that service nodes remain devoid of any information in instances where an editor fails to match a message. Within the binary access control scenario, the failure of a match prevents service nodes from distinguishing the fitness of the owner's or editor's identity.

Utility. The design of PIRB should prioritize flexibility and efficiency. It should be capable of accommodating one-way and bilateral access control, while also supporting

one policy for a batch of users. The core services within the PIRB framework, i.e., hash generation, match, and edit, should be executed with optimal efficiency. It is crucial that the PIRB system delivers these services accurately, with minimal computational and communication overheads.

Accountability. If a case of malicious editing behavior is identified within the PIRB system, appropriate penalties can be enforced through a traceable process. In the context of such misconduct, the consortium nodes possess the capability to ascertain the actual identity of an anonymous editor.

3. Preliminaries

3.1. Chameleon Hash with Ephemeral Trapdoors

The chameleon hash (CH) [32] empowers a message to be edited without changing its hash. Specifically, the public key and its trapdoor are generated first. To compute the hash of a message m, a randomness r is selected for m, and the hash h can be generated with (m, r). To replace m with m', an editor needs to obtain the trapdoor and compute a new randomness r' for m' where the hash of (m', r') is still h. To improve trapdoor security, the chameleon hash with ephemeral trapdoors (CHET) [33] was proposed. For each message m, a unique ephemeral trapdoor is generated to preserve the edit privilege from the leakage of the long-time trapdoor. The security of CHET is described in detail in [34]. In our constructions, we employ CHET to achieve the desirable property of redactability within the blockchain. The definition of CHET is presented as follows.

Definition 1. *CHET: The scheme CHET consists of five algorithms* (**PPGen**, **KTGen**, **Hash**, **Verify**, **Edit**).

PPGen $\lambda \rightarrow \mathsf{pp_{CHET}}$*: On inputting the security parameter* λ, **PPGen** *outputs the public parameter* $\mathsf{pp_{CHET}}$.

KTGen $\mathsf{pp_{CHET}} \rightarrow (\mathsf{pk_{CHET}}, \mathsf{ltd_{CHET}})$*: On inputting the public parameter* $\mathsf{pp_{CHET}}$, **KTGen** *outputs the public key* $\mathsf{pk_{CHET}}$ *and the long-term trapdoor* $\mathsf{ltd_{CHET}} = d_1$.

Hash $(\mathsf{pk_{CHET}}, m) \rightarrow (\mathsf{etd_{CHET}}, h, r)$*: On inputting the public key* $\mathsf{pk_{CHET}} = e$ *and the message* $m \in \mathcal{M}$, **Hash** *outputs the hash* $h = (h_1, h_2)$, *the randomness* $r = (r_1, r_2)$ *and the ephemeral trapdoor* $\mathsf{etd_{CHET}} = d_2$, *where* $\mathcal{M}$ *is the massage space. Specifically, for* $z \in \{1, 2\}$,

$$h_z = G_z(m) r_z^e \bmod N_z,$$

where $G_z : \{0,1\}^* \rightarrow \mathbb{Z}_{N_z}^*$, $N_z = p_z q_z$, $ed_z \equiv 1 \bmod (p_z - 1)(q_z - 1)$.

Verify $(\mathsf{pk_{CHET}}, m, h, r) \rightarrow V$*: On inputting the public key* $\mathsf{pk_{CHET}}$, *the message* m, *the hash* h *and the randomness* r, **Verify** *outputs the result* $V \in 0, 1$.

Edit $(\mathsf{ltd_{CHET}}, \mathsf{etd_{CHET}}, m, m', h, r) \rightarrow r'$*: On inputting the long-term trapdoor* $\mathsf{ltd_{CHET}} = d_1$, *the ephemeral trapdoor* $\mathsf{etd_{CHET}} = d_2$, *the message* m, *the edited message* m', *the hash* h *and the randomness* r, **Edit** *outputs the new randomness* $r' = (r_1', r_2')$. *Specifically, for* $z \in \{1, 2\}$,

$$r_z' = \left(h_z \left(G_z(m')\right)^{-1}\right)^{d_z} \bmod N_z,$$

to ensure that $G_z(m') r_z'^e \bmod N_z = G_z(m) r_z^e \bmod N_z = h_z$. *Therefore,* m *is edited to* m' *without changing the hash* h.

3.2. Identity-Based Encryption

Identity-based encryption (IBE) [35] allows users to set the identity-based access policy for their ciphertexts without the traditional public key so that the need for the certificate authority is mitigated. In our constructions, we utilize IBE to implement an access control mechanism for the trapdoors of CHET. The definition of IBE is presented as follows.

Definition 2. *IBE: The scheme IBE consists of four algorithms* (**Setup**, **KeyGen**, **Encrypt**, **Decrypt**).

Setup $\lambda \rightarrow (\mathsf{pp_{IBE}}, \mathsf{msk_{IBE}})$*: On inputting the security parameter* λ, **Setup** *outputs the public parameter* $\mathsf{pp_{IBE}} = (P, sP)$ *and the master secret key* $\mathsf{msk_{IBE}} = s$.

KeyGen $(\mathsf{pp_{IBE}}, \mathsf{msk_{IBE}}, ID) \rightarrow (\mathsf{pk_{CHET}}, \mathsf{sk_{CHET}})$*: On inputting the public parameter* $\mathsf{pp_{CHET}}$*, the master secret key* $\mathsf{msk_{IBE}}$ *and a user's identity* $ID \in \{0,1\}^*$*,* **KeyGen** *outputs the user's secret key* $\mathsf{sk_{ID}} = sQ_{ID}$*, where ID is used as a public key.*

Encrypt $(ID, m) \rightarrow c$*: On inputting the user's identity ID and the plaintext* $m \in \mathcal{M}$*,* **Encrypt** *outputs the ciphertext* $c \in \mathcal{C}$ *as*

$$c = (c_0, c_1) = (rP, m \oplus e(sP, Q_{ID})^r),$$

where $\mathcal{M}$ *is the plaintext space and* $\mathcal{C}$ *is the ciphertext space.*

Decrypt $(\mathsf{sk_{ID}}, c) \rightarrow m$*: On inputting the user's secret key* $\mathsf{sk_{ID}}$ *and the ciphertext c,* **Decrypt** *outputs the the plaintext* m' *as*

$$m' = c \oplus e(rP, sQ_{ID}) = m,$$

3.3. Polynomial Function

The polynomial function technique allows one policy to match with multiple roots. Specifically, to hide a secret t, the owner selects roots $x_1, x_2, \cdots, x_n$ as the keys of access control and computes

$$f(x) = r\prod_{i=1}^{n}(x - x_i) + t = \sum_{l=0}^{n} a_l x^l,$$

where r is a random value. Subsequently, the owner submits the coefficients $\{a_l\}_{l=0}^{n}$. Then, a user with the root $x_u \in \{x_i\}_{i=0}^{n}$ can compute

$$f(x_u) = \sum_{l=0}^{n} a_l x_u^l = r \prod_{i=1, i\neq u}^{n} (x_u - x_i) \cdot (x_u - x_u) + t = t,$$

to recover the secret t. Otherwise, the user cannot obtain t. In our constructions, we leverage the polynomial function technique to attain flexible policies, wherein one policy can be applied to a batch of users.

4. Proposed Schemes

In this section, we introduce the detailed design of the two schemes of PIRB, PIRB-I and PIRB-II. The notations are summarized in Table 2.

Table 2. Notations and descriptions.

Notation	Description	Notation	Description
Ψ	The description of the bilinear map.	$\mathsf{epk_{CHET,j}}$	Owner u_j's ephemeral public key ofCHET.
mpk	The master public key.	$\mathsf{etd_{CHET,j}}$	Owner u_j's ephemeral trapdoor of CHET.
msk	The master secret key.	$\mathsf{rsk_k}$	Editor u_k's edit re-encryption key.
$\mathcal{U}$	The set of all users.	m_j	Owner u_j's message.
$\mathcal{U}_i$	The set of user u_i's authorized editors or interested owners.	h_j	The hash of m_j.
N	The preset maximum policy size.	r_j	The randomness of m_j.
n_i	The number of elements in $\mathcal{U}_i$.	T_j	Owner u_j's match ciphertext.
ID_{u_i}	User u_i's identity.	$T_{j,0}$	Owner u_j's trapdoor ciphertext.
VID_{u_i}	User u_i's virtual identity.	$T_{j,1}$	Owner u_j's search ciphertext.
$\mathsf{sk_{IBE,i}}$	User u_i's secret key of IBE.	$T_{j,2}$	Owner u_j's return ciphertext.
$\mathsf{rsk_{IBE,i}}$	User u_i's re-encryption key of IBE.	K_k	Editor u_k's match key.
$\mathsf{pk_{CHET,j}}$	Owner u_j's long-term public key of CHET.	$R_{k,j}$	Editor u_k's match result with owner u_j.
$\mathsf{ltd_{CHET,j}}$	Owner u_j's long-term trapdoor of CHET.	R_k	The set of editor u_k's match results.

4.1. Proposed PIRB-I

4.1.1. Main Idea

PIRB-I is a privacy-preserving identity-based redactable blockchain scheme with accountability and one-way access control. The consortium nodes first exchange the identity-based secret keys to the users and the associated re-encryption keys to the service nodes simultaneously as the basis of policy generation. Then, a number of knowledge owners generate the match ciphertext with identity-based policies while computing the

hashes and submitting the messages to the service nodes. When a knowledge editor requests the edit privilege, the editor computes the match key and submits it to the service nodes. After receiving the match key, the service nodes operate a match with the re-encryption key and return the match result to the editor. Finally, the editor decrypts the result and computes the new randomness to edit the message without changing the hash.

4.1.2. Detailed Construction

PIRB-I consists of seven phases, i.e., **Setup**, **KeyGen**, **Hash**, **Verify**, **Match**, **Edit**, **Trace**. The workflow of PIRB-I is shown in Figure 2.

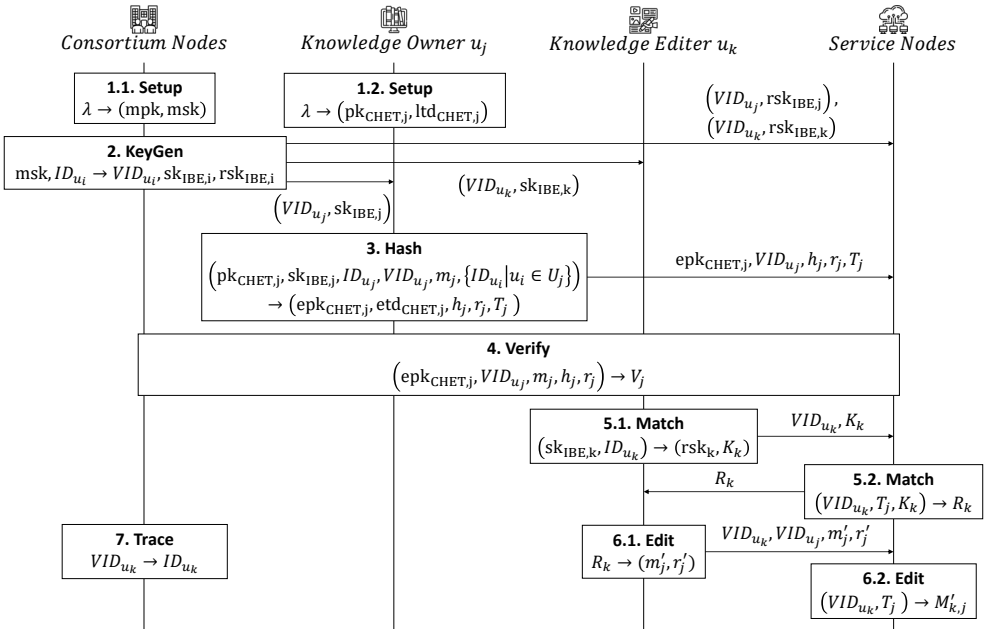

Figure 2. Workflow of PIRB-I.

Setup $\lambda \rightarrow (\mathsf{mpk}, \mathsf{msk}, \mathsf{pk}_{\mathsf{CHET,j}}, \mathsf{ltd}_{\mathsf{CHET,j}})$: The consortium nodes cooperate to generate a description of the bilinear map $\Psi = (p, g, \mathbb{G}, \mathbb{G}_T, e)$, where $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$, and select two hash functions $H_1[\cdot] : \{0,1\}^* \rightarrow \mathbb{Z}_p$ and $H_2[\cdot] : \mathbb{G}_T \rightarrow \{0,1\}^*$. Subsequently, the consortium nodes select a random value $s \in \mathbb{Z}_p$ as the master secret key msk and select $(g^{\frac{1}{s}}, H_1[\cdot], H_2[\cdot], \Psi)$ as the master public key mpk.

Simultaneously, every knowledge owner generates a long-term trapdoor for chameleon hash with ephemeral trapdoors (CHET). Specifically, knowledge owner u_j selects primes $e_j, p_{j,1}, q_{j,1}$ and set $N_{j,1} = p_{j,1}q_{j,1}$. Then, u_j computes $d_{j,1}$ that $e_j d_{j,1} \equiv 1 \bmod (p_{j,1} - 1)(q_{j,1} - 1)$ as the long-term trapdoor of CHET $\mathsf{ltd}_{\mathsf{CHET,j}}$, chooses a hash function $G_{j,1} : \{0,1\}^* \rightarrow \mathbb{Z}^*_{N_{j,1}}$ and selects $(N_{j,1}, e_j, G_{j,1})$ as the long-term public key $\mathsf{pk}_{\mathsf{CHET,j}}$.

KeyGen $(\mathsf{msk}, \{ID_{u_i} | u_i \in \mathcal{U}\}) \rightarrow (\{VID_{u_i} | u_i \in \mathcal{U}\}, \{\mathsf{sk}_{\mathsf{IBE,i}} | u_i \in \mathcal{U}\}, \{\mathsf{rsk}_{\mathsf{IBE,i}} | u_i \in \mathcal{U}\})$: For each user in the user set $\mathcal{U}$, including each knowledge owner, the consortium nodes generate a secret key. Specifically, for user $u_i \in \mathcal{U}$ with identity ID_{u_i}, the consortium nodes generate a virtual identity VID_{u_i} randomly. Subsequently, the consortium nodes select random value $s_{i,0}$ as u_i's re-encryption key $\mathsf{rsk}_{\mathsf{IBE,i}}$ and compute $g^{s_{i,0}sH_1(ID_{u_i})}$ as u_i's secret key $\mathsf{sk}_{\mathsf{IBE,i}}$. Then, the consortium nodes send $(VID_{u_i}, \mathsf{sk}_{\mathsf{IBE,i}})$ to u_i and $(VID_{u_i}, \mathsf{rsk}_{\mathsf{IBE,i}})$ to the service nodes via a secure channel.

Hash $(\mathsf{pk}_{\mathsf{CHET,j}}, \mathsf{sk}_{\mathsf{IBE,j}}, ID_{u_j}, VID_{u_j}, m_j, \{ID_{u_i} | u_i \in \mathcal{U}_j\}) \rightarrow (\mathsf{epk}_{\mathsf{CHET,j}}, \mathsf{etd}_{\mathsf{CHET,j}}, h_j, r_j, T_j)$: For a message of knowledge m_j on the blockchain, the knowledge owner u_j computes the hash value h_j and sets the access policy. Note that the knowledge m_j can be stored either in plaintext or ciphertext, allowing the owner u_j to employ ciphertext as a means to safeguard data privacy. Specifically, u_j selects primes $p_{j,2}, q_{j,2}$ and sets $N_{j,2} = p_{j,2}q_{j,2}$. Compute $d_{j,2}$ that $e_j d_{j,2} \equiv 1 \bmod (p_{j,2} - 1)(q_{j,2} - 1)$ as the ephemeral trapdoor of CHET $\mathsf{etd}_{\mathsf{CHET,j}}$. Choose

a hash function $G_{j,2} : \{0,1\}^* \to \mathbb{Z}^*_{N_{j,2}}$ and select $(e_j, N_{j,1}, G_{j,1}, N_{j,2}, G_{j,2})$ as the ephemeral public key $\mathsf{epk}_{\mathsf{CHET,j}}$. Subsequently, u_j generates the hash of m_j as $h_j = (h_{j,1}, h_{j,2})$, and for $z \in \{1,2\}$,

$$h_{j,z} = G_{j,z}\Big(VID_{u_j}, m_j, G_{j,1}, N_{j,1}, G_{j,2}, N_{j,2}\Big) r_{j,z}^{e_j} \bmod N_{j,z},$$

where $r_{j,1}, r_{j,2}$ are random values and $r_j = (r_{j,1}, r_{j,2})$. To set the access policy, the knowledge owner u_j selects a random value $t \in \mathbb{Z}_p$ and calculates the trapdoor ciphertext $T_{j,0} = (T_{j,0,1}, T_{j,0,2}) = (d_{j,1} \oplus G_{j,1}(e(g,g)^t), d_{j,2} \oplus G_{j,2}(e(g,g)^t))$. Subsequently, u_j selects a list $\mathcal{U}_j = \{u_1, u_2, \cdots, u_n\}$ with n members who have the adapting right and expands the number of elements to N with fuzzy identities as $\mathcal{U}'_j = \{u_1, u_2, \cdots, u_n, u'_{n+1}, \cdots, u'_N\}$, $\mathcal{U}''_j = \{u_1, u_2, \cdots, u_n, u''_{n+1}, \cdots, u''_N\}$ before the generation of polynomial functions. Note that to weaken the correlation between the two functions, the fuzzy identities of each polynomial function are unique. In addition, the fuzzy identities are selected to avoid affecting the **Match** phase. Then, u_j generates

$$f_1(x) = r_1 \prod_{i=1, u'_i \in \mathcal{U}'_j}^{N} \Big(x - H_1\Big(ID_{u'_i}\Big)\Big) = \sum_{l=0}^{N} a_l x^l,$$

$$f_2(x) = r_2 \prod_{i=1, u''_i \in \mathcal{U}''_j}^{N} \Big(x - H_1\Big(ID_{u''_i}\Big)\Big) + t = \sum_{l=0}^{N} b_l x^l,$$

where $r_1, r_2 \in \mathbb{Z}_p$ are random values.

Then, u_j computes the search ciphertext $T_{j,1}$ as

$$\begin{aligned} &T_{j,1} = (T_{j,1,N}, T_{j,1,N-1}, \cdots, T_{j,1,1}, T_{j,1,0}), \\ &T_{j,1,l} = g^{\frac{a_l}{s}}, 1 \le l \le N, \\ &T_{j,1,0} = g^{a_0}, \end{aligned}$$

and the return ciphertext $T_{j,2}$ as

$$\begin{aligned} &T_{j,2} = (T_{j,2,N}, T_{j,2,N-1}, \cdots, T_{j,2,1}, T_{j,2,0}), \\ &T_{j,2,l} = g^{\frac{b_l}{s}}, 1 \le l \le N, \\ &T_{j,1,0} = g^{b_0}, \end{aligned}$$

with mpk. Then, u_j sets the match ciphertext $T_j = (T_{j,0}, T_{j,1}, T_{j,2})$.

Finally, u_j sends $(\mathsf{epk}_{\mathsf{CHET,j}}, VID_{u_j}, h_j, r_j, T_j)$ to the service nodes. Note that with the message above, the service nodes cannot obtain ID_{u_j}.

Verify $(\mathsf{epk}_{\mathsf{CHET,j}}, VID_{u_j}, m_j, h_j, r_j) \to V_j$: After the service nodes receive the message, every node can verify the hash. Specifically, with $\mathsf{epk}_{\mathsf{CHET,j}}$, the node checks $h_{j,1} \stackrel{?}{=} G_{j,1}(VID_{u_j}, m_j, G_{j,1}, N_{j,1}, G_{j,2}, N_{j,2}) r_{j,1}^{e_j}$ and $h_{j,2} \stackrel{?}{=} G_{j,2}(VID_{u_j}, m_j, G_{j,1}, N_{j,1}, G_{j,2}, N_{j,2}) r_{j,2}^{e_j}$. If all checks hold, the node returns the result $V_j = 1$ and otherwise returns $V_j = 0$.

Match $(VID_{u_k}, T_j, K_k) \to R_k$: To find the editable message for u_k on the blockchain, the user u_k sends a request to the service nodes, and the service nodes return the message to u_k. Specifically, with $\mathsf{sk}_{\mathsf{IBE,k}} = g^{s_{k,0} s H_1(ID_{u_k})}$, u_k calculates the match key K_k as

$$\begin{aligned} &K_k = (K_{k,N}, K_{k,N-1}, \cdots, K_{k,1}, K_{k,0}), \\ &K_{k,m} = g^{s_k s_{k,0} s H_1(ID_{u_k})^m}, 1 \le m \le N, \\ &K_{k,0} = g^{s_k}, \end{aligned}$$

where s_k is a random value, and it sets the edit re-encryption key $\mathsf{rsk_k} = s_k$. Then, u_k sends K_k to the service nodes. Note that with K_k, the service nodes cannot obtain ID_{u_j}. After receiving K_k, with $T_{j,1}$, the service nodes calculate

$$M_{k,j} = \prod_{l=1}^{N} e\left(T_{j,1,l}, K_{k,l}^{\frac{1}{s_{k,0}}}\right) \cdot e\left(T_{j,1,0}, K_{k,0}\right)$$

to match u_k with u_j's message and then check $M_{j,k} \stackrel{?}{=} e(g,g)^0$. If $u_k \in U_j$, $M_{j,k} = e(g,g)^0$; otherwise, the service nodes cannot obtain any information from $M_{j,k}$. Then, with $T_{j,2}$, the service nodes calculate

$$\begin{aligned} R_{k,j,0} =& \prod_{l=1}^{N} e\left(T_{j,2,l}, K_{k,l}^{\frac{1}{s_{k,0}}}\right) \cdot e\left(T_{j,2,0}, K_{k,0}\right) \\ =& e(g,g)^{s_k t}, \end{aligned}$$

and select $R_{k,j} = (R_{k,j,0}, \mathsf{epk_{CHET,j}}, VID_{u_j}, m_j, h_j, r_j, T_j)$. Finally, the service nodes return the set of match results $R_k = \{R_{k,j} | M_{k,j} = e(g,g)^0\}$ to u_k.

Edit $(\mathsf{rsk_k}, R_k) \rightarrow (m'_j, r'_j)$: Receiving the set of match results from the service nodes, u_k obtains the edit privilege. Our constructions provide support for both on-chain storage and hybrid storage approaches. In the interest of conciseness, we will focus on hybrid storage. Specifically, u_k first verifies the hash as the phase of **Verify**. If $V_j = 1$, with $R_{k,j,0} = e(g,g)^{s_k t}$ and the edit re-encryption key $\mathsf{rsk_k} = s_k$, u_k computes $(e(g,g)^{s_k t})^{\frac{1}{s_k}} = e(g,g)^t$ and $T_{j,0,1} \oplus G_{j,1}(e(g,g)^t) = d_{j,1}, T_{j,0,2} \oplus G_{j,2}(e(g,g)^t) = d_{j,2}$. Subsequently, for $z \in \{1,2\}$, u_k computes

$$r'_{j,z} = \left(h_{j,z}\left(G_{j,z}\left(VID_{u_j}, m'_j, G_{j,1}, N_{j,1}, G_{j,2}, N_{j,2}\right)\right)^{-1}\right)^{d_{j,z}} \bmod N_{j,z},$$

and sets $r'_j = (r'_{j,1}, r'_{j,2})$ to change m_j to m'_j and then checks $h_{j,1} \stackrel{?}{=} G_{j,1}(VID_{u_j}, m'_j, G_{j,1}, N_{j,1}, G_{j,2}, N_{j,2}) r'^{\,e_j}_{j,1}$ and $h_{j,2} \stackrel{?}{=} G_{j,2}(VID_{u_j}, m'_j, G_{j,1}, N_{j,1}, G_{j,2}, N_{j,2}) r'^{\,e_j}_{j,2}$. Next, u_k sends $(VID_{u_j}, VID_{u_k}, m'_j, r'_j)$ to the service nodes by encapsulating it within a transaction. If all checks hold, the service nodes submit the updated data $(VID_{u_j}, VID_{u_k}, m'_j, r'_j)$ to the blockchain, ensuring that the original data are modified while preserving the integrity of the hash value.

Trace $VID_{u_k} \rightarrow ID_{u_k}$: If an editor is found to display malicious behaviors, the consortium nodes will reveal their actual identity to the public for penalties. Specifically, if the malicious behaviors of u_k are found, with the virtual identity VID_{u_k}, the consortium nodes will find the associated actual identity ID_{u_k} and reveal it to the public to stop the service to u_k and impose penalties on the editor.

4.2. Proposed PIRB-II

4.2.1. Main Idea

PIRB-I addresses the challenge in one-way access control scenarios; however, the absence of predetermined owner selection may lead to the excessive influx of messages for editors. In light of this, we introduce PIRB-II, which empowers editors to establish policies for owner selection. This refinement aims to mitigate the issue of message overload for editors, enhancing the overall system's efficiency and functionality. To enable bilateral access control, PIRB-II is different in the **Hash** and **Match** phases from PIRB-I. Specifically, the match key of the owner and the match ciphertext of the editor are also needed in the **Match** phase. To preserve the match privacy, the owner and the editor, respectively, generate the match ciphertext and key with a random value.

4.2.2. Detailed Construction

PIRB-II consists of seven phases, i.e., **Setup**, **KeyGen**, **Hash**, **Verify**, **Match**, **Edit**, **Trace**. We highlight the **Hash** and **Match** phases in PIRB-II, which are different from those in PIRB-I. The workflow of PIRB-II is shown in Figure 3.

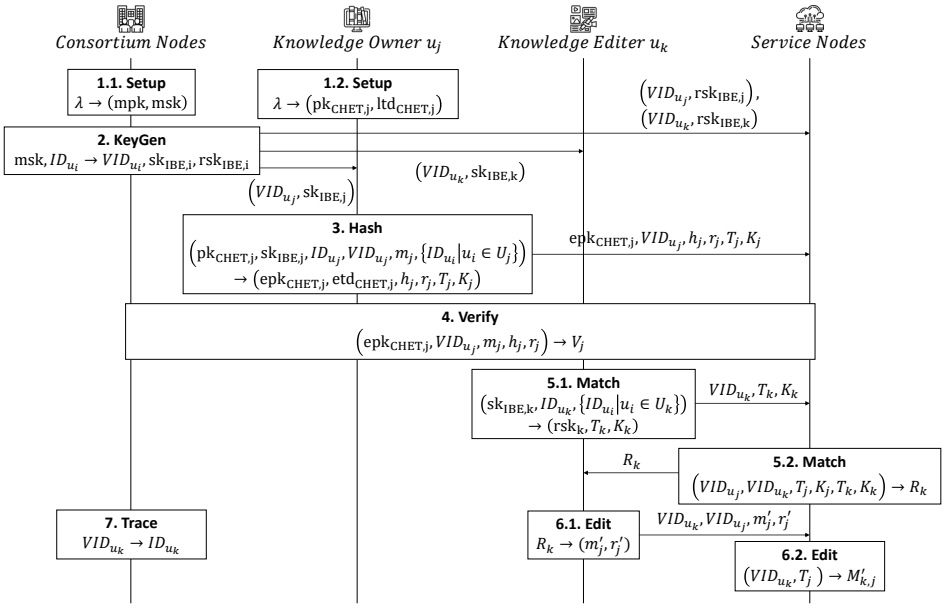

Figure 3. Workflow of PIRB-II.

Setup $\lambda \rightarrow (\mathsf{mpk}, \mathsf{msk}, \mathsf{pk_{CHET,j}}, \mathsf{ltd_{CHET,j}})$: The phase of **Setup** is the same as PIRB-I.

KeyGen $(\mathsf{msk}, \{ID_{u_i} | u_i \in \mathcal{U}\}) \rightarrow (\{VID_{u_i} | u_i \in \mathcal{U}\}, \{\mathsf{sk_{IBE,i}} | u_i \in \mathcal{U}\}, \{\mathsf{rsk_{IBE,i}} | u_i \in \mathcal{U}\})$: The phase of **KeyGen** is the same as PIRB-I.

Hash $(\mathsf{pk_{CHET,j}}, \mathsf{sk_{IBE,k}}, ID_{u_j}, VID_{u_j}, m_j, \{ID_{u_i} | u_i \in \mathcal{U}_j\}) \rightarrow (\mathsf{epk_{CHET,j}}, \mathsf{etd_{CHET,j}}, h_j, r_j, T_j, K_j)$: For a message of knowledge m_j in plaintext or ciphertext on the blockchain, the knowledge owner u_j computes the hash value h_j and sets the access policy. Specifically, as in PIRB-I, u_j selects primes $p_{j,2}, q_{j,2}$, computes $N_{j,2}, d_{j,2}$ and chooses a hash function $G_{j,2}$. Then, u_j selects $\mathsf{epk_{CHET,j}} = (e_j, N_{j,1}, G_{j,1}, N_{j,2}, G_{j,2})$ and $\mathsf{etd_{CHET,j}} = d_{j,2}$. Subsequently, u_j generates the hash h_j and sets $r_j = (r_{j,1}, r_{j,2})$. To set the access policy, the knowledge owner u_j selects a random value $t \in \mathbb{Z}_p$, calculates the trapdoor ciphertext $T_{j,0}$, selects a list $\mathcal{U}_j$ and expands the number of elements to N with fuzzy identities as $\mathcal{U}'_j, \mathcal{U}''_j$ as in PIRB-I before the generation of polynomial functions. Then, u_j generates

$$f_{j,1}(x) = r_1 \prod_{i=1, u'_i \in \mathcal{U}'_j}^{N} \left(x - H_1\left(ID_{u'_i}\right)\right) - s_j = \sum_{l=0}^{N} a_l x^l,$$

$$f_{j,2}(x) = r_2 \prod_{i=1, u''_i \in \mathcal{U}''_j}^{N} \left(x - H_1\left(ID_{u''_i}\right)\right) + t - s_j = \sum_{l=0}^{N} b_l x^l,$$

where $r_1, r_2, s_j \in \mathbb{Z}_p$ are random values.

Then, u_j computes the search ciphertext $T_{j,1}$ and the return ciphertext $T_{j,2}$ as in PIRB-I, with mpk and sets the match ciphertext $T_j = (T_{j,0}, T_{j,1}, T_{j,2})$. Subsequently, with $\mathsf{sk_{IBE,k}}$, u_j generates the match key K_j as

$$K_j = (K_{j,N}, K_{j,N-1}, \cdots, K_{j,1}, K_{j,0}),$$
$$K_{j,l} = g^{s_j s_{j,0} s H_1(ID_{u_j})^l}, 1 \leq l \leq N,$$
$$K_{j,0} = g^{s_j},$$

Finally, u_j submits $(\mathsf{epk_{CHET,j}}, VID_{u_j}, h_j, r_j, T_j, K_j)$ to the blockchain.

Verify $(\mathsf{epk_{CHET,j}}, VID_{u_j}, m_j, h_j, r_j) \rightarrow V_j$: The phase of **Verify** is the same as in PIRB-I.

Match $(VID_{u_k}, VID_{u_j}, T_j, K_j, T_k, K_k) \rightarrow R_k$: To find the editable message for u_k on the blockchain, the user u_k sends a request to the service nodes, and the service nodes return the message to u_k. Specifically, u_k selects a list $\mathcal{U}_k = \{u_{k,1}, u_{k,2}, \cdots, u_{k,n_k}\}$ with n_k interested owners and expands the number of elements to N with fuzzy identities as $\mathcal{U}'_k = \{u_{k,1}, u_{k,2}, \cdots, u_{k,n_k}, u'_{k,n_k+1}, \cdots, u'_{k,N}\}$ before the generation of polynomial functions. Then, u_k generates

$$f_k(x) = r_3 \prod_{i=1, u'_i \in \mathcal{U}'_k}^{N} \left(x - H_1\left(ID_{u'_i}\right)\right) + s_k = \sum_{m=0}^{N} c_m x^m,$$

where r_3, s_k are random values, and it computes the match ciphertext

$$\begin{aligned} T_k &= (T_{k,N}, T_{k,N-1}, \cdots, T_{k,1}, T_{k,0}), \\ T_{k,m} &= g^{\frac{c_m}{s}}, 1 \le m \le N, \\ T_{k,0} &= g^{a_0}, \end{aligned}$$

with mpk. Subsequently, with $\mathsf{sk}_{\mathsf{IBE,k}}$, u_k calculates the match key K_k as in PIRB-I and sets the edit re-encryption key $\mathsf{rsk}_{\mathsf{k}} = s_k$. Then, u_k sends (VID_{u_k}, T_k, K_k) to the service nodes. With $\mathsf{rsk}_{\mathsf{IBE,k}} = s_{k,0}$ and $\mathsf{rsk}_{\mathsf{IBE,j}} = s_{j,0}$, the service nodes calculate

$$M_{k,j} = \prod_{l=1}^{N} e\left(T_{j,1,l}, K_{k,l}^{\frac{1}{s_{k,0}}}\right) \cdot e\left(T_{j,1,0}, K_{k,0}\right) \cdot \prod_{m=1}^{N} e\left(T_{k,l}, K_{j,l}^{\frac{1}{s_{j,0}}}\right) \cdot e\left(T_{k,0}, K_{j,0}\right),$$

with $T_{j,1}$ to match u_k with u_j's message and check $M_{j,k} \stackrel{?}{=} e(g,g)^{s_k(-s_j)+s_j s_k} = e(g,g)^0$. If $M_{j,k} = e(g,g)^0$, $u_k \in \mathcal{U}_j$ and $u_j \in \mathcal{U}_k$; otherwise, with the random values s_j, s_k, the service nodes cannot obtain any information from $M_{j,k}$. Then, with $T_{j,2}$, the service nodes calculate

$$\begin{aligned} R_{k,j,0} =& \prod_{l=1}^{N} e\left(T_{j,2,l}, K_{k,l}^{\frac{1}{s_{k,0}}}\right) \cdot e\left(T_{j,1,0}, K_{k,0}\right) \cdot \prod_{m=1}^{N} e\left(T_{k,m}, K_{j,m}^{\frac{1}{s_{j,0}}}\right) \cdot e\left(T_{k,0}, K_{j,0}\right) \\ =& e(g,g)^{s_k\left(-s_j\right)+s_k t} \cdot e(g,g)^{s_j s_k} = e(g,g)^{s_k t}, \end{aligned}$$

and select $R_{k,j} = (R_{k,j,0}, \mathsf{epk}_{\mathsf{CHET,j}}, VID_{u_j}, m_j, h_j, r_j, T_j)$. Finally, the service nodes return the set of match results $R_k = \{R_{k,j} | M_{k,j} = e(g,g)^0\}$ to u_k.

Edit $(\mathsf{rsk}_{\mathsf{k}}, R_k) \rightarrow (m'_j, r'_j)$: The phase of **Edit** is the same as in PIRB-I.

Trace $VID_{u_k} \rightarrow ID_{u_k}$: The phase of **Trace** is the same as in PIRB-I.

5. Theoretical Analysis

5.1. Correctness Analysis

We provide a correctness analysis for the **Match** and **Edit** phases in PIRB.

Theorem 1. *In PIRB-I, if and only if $(u_k \in \mathcal{U}_j)$ holds, the service node obtains a successful match between the owner u_j and the editor u_k. Otherwise, the service node cannot obtain any meaningful match result.*

Proof. If $(u_k \in \mathcal{U}_j)$, ID_{u_k} is used to generate the polynomial function f_1. Then, in the **Match** phase, with $T_{j,1}$, K_k and $\mathsf{rsk}_{\mathsf{IBE,k}} = s_{k,0}$, the service nodes compute

$$\begin{aligned} M_{k,j} =& \prod_{l=1}^{N} e\left(T_{j,1,l}, K_{k,l}^{\frac{1}{s_{k,0}}}\right) \cdot e\left(T_{j,1,0}, K_{k,0}\right) \\ =& \prod_{l=1}^{N} e\left(g^{\frac{a_l}{s}}, g^{s_k s H_1\left(ID_{u_k}\right)^l}\right) \cdot e\left(g^{a_0}, g^{s_k}\right) \\ =& e(g,g)^{s_k\left(\sum_{l=0}^{N} a_l H_1\left(ID_{u_k}\right)^l\right)}, \end{aligned}$$

and obtain $M_{k,j} = e(g,g)^0$. Similarly, if $(u_k \in \mathcal{U}_j)$, ID_{u_k} is used to generate the polynomial function f_2. Then, with $T_{j,2}$, K_k and $\mathsf{rsk_{IBE,k}} = s_{k,0}$, the service nodes similarly compute

$$\begin{aligned} R_{k,j,0} =& \prod_{l=1}^{N} e\left(T_{j,2,l}, K_{k,l}^{\frac{1}{s_{k,0}}}\right) \cdot e\left(T_{j,2,0}, K_{k,0}\right) \\ =& \prod_{l=1}^{N} e\left(g^{\frac{b_l}{s}}, g^{s_k s H_1\left(ID_{u_k}\right)^l}\right) \cdot e\left(g^{b_0}, g^{s_k}\right) \\ =& e(g,g)^{s_k\left(\sum_{l=0}^{N} b_l H_1\left(ID_{u_k}\right)^l\right)} = e(g,g)^{s_k t}, \end{aligned}$$

and return the match result to the editor u_k. Otherwise, the service nodes cannot obtain $M_{k,j} = e(g,g)^0$ or $R_{k,j,0} = e(g,g)^{s_k t}$. Thus, Theorem 1 is proven. □

Theorem 2. *In PIRB-II, if and only if $(u_k \in \mathcal{U}_j) \wedge (u_j \in \mathcal{U}_k)$ holds, the service nodes obtain a successful match between the owner u_j and the editor u_k. Otherwise, the service nodes cannot obtain any meaningful match result.*

Proof. Similar to Theorem 1, if $(u_k \in \mathcal{U}_j) \wedge (u_j \in \mathcal{U}_k)$, ID_{u_k} is used to generate the polynomial functions $f_{j,1}$ and $f_{j,2}$, and ID_{u_j} is used to generate the polynomial function f_k. Then, in the **Match** phase, with $T_{j,1}$, K_k, $\mathsf{rsk_{IBE,k}} = s_{k,0}$ and $\mathsf{rsk_{IBE,j}} = s_{j,0}$, the service nodes compute

$$\begin{aligned} M_{k,j} =& \prod_{l=1}^{N} e\left(T_{j,1,l}, K_{k,l}^{\frac{1}{s_{k,0}}}\right) \cdot e\left(T_{j,1,0}, K_{k,0}\right) \cdot \prod_{m=1}^{N} e\left(T_{k,m}, K_{j,m}^{\frac{1}{s_{j,0}}}\right) \cdot e\left(T_{k,0}, K_{j,0}\right) \\ =& \prod_{l=1}^{N} e\left(g^{\frac{a_l}{s}}, g^{s_k s H_1\left(ID_{u_k}\right)^l}\right) \cdot e\left(g^{a_0}, g^{s_k}\right) \cdot \prod_{m=1}^{N} e\left(g^{\frac{c_m}{s}}, g^{s_j s H_1\left(ID_{u_j}\right)^m}\right) \cdot e\left(g^{c_0}, g^{s_j}\right) \\ =& e(g,g)^{s_k\left(\sum_{l=0}^{N} a_l H_1\left(ID_{u_k}\right)^l\right) + s_j\left(\sum_{m=0}^{N} c_m H_1\left(ID_{u_j}\right)^m\right)} \\ =& e(g,g)^{s_k(-s_j)} \cdot e(g,g)^{s_j s_k} = e(g,g)^0. \end{aligned}$$

Then, with $T_{j,2}$, K_k, $\mathsf{rsk_{IBE,k}} = s_{k,0}$ and $\mathsf{rsk_{IBE,j}} = s_{j,0}$, the service nodes similarly compute

$$\begin{aligned} R_{k,j,0} =& \prod_{l=1}^{N} e\left(T_{j,2,l}, K_{k,l}^{\frac{1}{s_{k,0}}}\right) \cdot e\left(T_{j,2,0}, K_{k,0}\right) \cdot \prod_{m=1}^{N} e\left(T_{k,m}, K_{j,m}^{\frac{1}{s_{j,0}}}\right) \cdot e\left(T_{k,0}, K_{j,0}\right) \\ =& \prod_{l=1}^{N} e\left(g^{\frac{b_l}{s}}, g^{s_k s H_1\left(ID_{u_k}\right)^l}\right) \cdot e\left(g^{b_0}, g^{s_k}\right) \cdot \prod_{m=1}^{N} e\left(g^{\frac{c_m}{s}}, g^{s_j s H_1\left(ID_{u_j}\right)^m}\right) \cdot e\left(g^{c_0}, g^{s_j}\right) \\ =& e(g,g)^{s_k\left(\sum_{l=0}^{N} b_l H_1\left(ID_{u_k}\right)^l\right) + s_j\left(\sum_{m=0}^{N} c_m H_1\left(ID_{u_j}\right)^m\right)} \\ =& e(g,g)^{s_k\left(-s_j\right) + s_k t} \cdot e(g,g)^{s_j s_k} = e(g,g)^{s_k t}, \end{aligned}$$

and return the match result to the editor u_k. Otherwise, the service nodes cannot obtain $M_{k,j} = e(g,g)^0$ or $R_{k,j,0} = e(g,g)^{s_k t}$. Thus, Theorem 2 is proven. □

Theorem 3. *In PIRB-I and PIRB-II, if and only if the editor u_k receives the result of a successful match with the owner u_j and has the edit re-encryption key $\mathsf{rsk_k} = s_k$, u_k obtains u_j's trapdoors (i.e., the long-term trapdoor and the ephemeral trapdoor) of CHET and edits the message m_j to m'_j. Otherwise, the editor cannot obtain the edit privilege.*

Proof. In both PIRB-I and PIRB-II, if the service nodes obtain a successful match between the owner u_j and the editor u_k, the editor u_k receives $e(g,g)^{s_k t}$ from the match result. If u_k has the edit re-encryption key $\mathsf{rsk_k} = s_k$, u_k computes

$$(e(g,g)^{s_k t})^{\frac{1}{s_k}} = e(g,g)^t.$$

Moreover, with the trapdoor ciphertext $T_{j,0}$, u_k computes

$$T_{j,0,1} \oplus G_{j,1}(e(g,g)^t) = d_{j,1}, T_{j,0,2} \oplus G_{j,2}(e(g,g)^t) = d_{j,2},$$

to obtain the long-term trapdoor $d_{j,1}$ and the ephemeral trapdoor $d_{j,2}$. To edit the message m_j to m'_j, with $d_{j,1}$ and $d_{j,2}$, for $z \in \{1,2\}$, u_k computes

$$r'_{j,z} = \left(h_{j,z}\left(G_{j,z}\left(VID_{u_j}, m'_j, G_{j,1}, N_{j,1}, G_{j,2}, N_{j,2}\right)\right)^{-1}\right)^{d_{j,z}} \bmod N_{j,z},$$

so that

$$\begin{aligned} r'^{\,e_j}_{j,z} &= \left(\left(h_{j,z}\left(G_{j,z}\left(VID_{u_j}, m'_j, G_{j,1}, N_{j,1}, G_{j,2}, N_{j,2}\right)\right)^{-1}\right)^{d_{j,z}}\right)^{e_j} \bmod N_{j,z} \\ &= h_{j,z}\left(G_{j,z}\left(m'_j, G_{j,1}, N_{j,1}, G_{j,2}, N_{j,2}\right)\right)^{-1} \bmod N_{j,z}, \\ G_{j,z}\left(m'_j, G_{j,1}, N_{j,1}, G_{j,2}, N_{j,2}\right) &r'^{\,e_j}_{j,z} \bmod N_{j,z} = h_{j,z}. \end{aligned}$$

Thus, u_k obtains valid randomness r' to edit m_j to m'_j without changing the hash $h_j = (h_{j,1}, h_{j,2})$. Otherwise, u_k cannot obtain the trapdoors of CHET and the valid randomness r'. Thus, Theorem 3 is proven. □

5.2. Security Analysis

In the security analysis, we prove the privacy preservation (i.e., trapdoor, identity, policy, and match privacy) and accountability of PIRB.

Theorem 4. *Our encryption approach is secure under the chosen-plaintext attack model.*

Proof. As shown in Algorithm 1, we provide an experiment between challenger $\mathcal{C}$ and adversary $\mathcal{A}$. Based on the experiment, we define security under the CPA model as

$$\mathsf{Adv}_{\mathcal{A}}^{IND-CPA}(\lambda) = |Pr[b' = b|\mathcal{A}(DB_0, DB_1, T)] - \frac{1}{2}| \leq \epsilon,$$

where ϵ is negligible.

Algorithm 1 An experiment between the adversary and the challenger

Setup: The challenger $\mathcal{C}$ selects a random value $s \in \mathbb{Z}_p$ to denote the master secret key and initializes $(g^{\frac{1}{s}}, H_1[\cdot], H_2[\cdot], \Psi)$ to denote the master public key.
Phase1: The adversary $\mathcal{A}$ applies for the secret key from $\mathcal{C}$. Then, $\mathcal{C}$ selects a random value $s_{A,0} \in \mathbb{Z}_p$ to denote the re-encryption key and generates and then publishes the secret key $g^{s_{A,0} s H_1(ID_A)}$ without $s_{A,0}$.
Challenge: $\mathcal{A}$ chooses the secret key $g^{s_{A,0} s H_1(ID_A)}$ and two databases $DB_0 = (t_{0,1}, t_{0,2}, \cdots, t_{0,n})$ and $DB_1 = (t_{1,1}, t_{1,2}, \cdots, t_{1,n})$, where $\{t_{z,i}\}_{i=0}^n \subseteq \mathbb{Z}_p$ for $z \in \{0,1\}$. $\mathcal{C}$ tosses a random binary coin $b \in \{0,1\}$ beyond the purview of $\mathcal{A}$. Then, $\mathcal{C}$ selects $\{r_i\}_{i=0}^n \subseteq \mathbb{Z}_p$ and generates the polynomial coefficients with the identity policy including the adversary $\mathcal{A}$'s identity. Subsequently, $\mathcal{C}$ encrypts $\{t_{b,i}\}_{i=0}^n \subseteq \mathbb{Z}_p$ to the ciphertexts $T = \{T_i\}_{i=0}^n$ and publishes the ciphertexts.
Guess: $\mathcal{A}$ submits the guess b'.

Assume that $DB_0 = (t_{0,1}, t_{0,2}, \cdots, t_{0,n})$ is the encrypted database. In the **Challenge** phase, for $i \in \{1,2,\cdots,n\}$, the challenger $\mathcal{C}$ picks $r_i \in \mathbb{Z}_p$ and selects the list $\mathcal{U}' = \{u_1, u_2, \cdots, u_n, u'_{n+1}, \cdots, u'_N\}$ where $u \in \mathcal{U}'$. Then, $\mathcal{C}$ generates the polynomial coefficients $\{b_{i,l}\}_{l=0}^N$ where

$$r_i \prod_{l=1, u'_l \in \mathcal{U}'}^{N} \left(x - H_1\left(ID_{u'_l}\right)\right) + t_{0,i} = \sum_{l=0}^{N} b_{i,l} x^l.$$

Subsequently, $\mathcal{C}$ generates

$$T_i = (T_{i,N}, T_{i,N-1}, \cdots, T_{i,1}, T_{i,0}),$$
$$T_{i,l} = g^{\frac{b_l}{s}}, 1 \le l \le N,$$
$$T_{i,0} = g^{b_0},$$

and publishes the ciphertexts.

To estimate the encrypted database, the adversary $\mathcal{A}$ selects $s_A \in \mathbb{Z}_p$ and computes

$$K = (K_N, K_{N-1}, \cdots, K_1, K_0),$$
$$K_l = g^{s_A s_{A,0} s H_1(ID_A)^l}, 1 \le l \le N,$$
$$K_0 = g^{s_A}.$$

Then, the adversary $\mathcal{A}$ attempts to decrypt the encryption. However, without the re-encryption key $s_{A,0}$, $\mathcal{A}$ must set $s'_{A,0}$ as a random value and compute

$$\begin{aligned} R' =& \prod_{l=1}^{N} e\left(T_{j,2,l}, K_{k,l}^{\frac{1}{s'_{A,0}}}\right) \cdot e\left(T_{j,2,0}, K_{k,0}\right) \\ =& e(g,g)^{s_A\left(\frac{s_{A,0}}{s'_{A,0}}\left(\sum_{l=1}^{N} b_l H_1(ID_A)^l\right)+b_0\right)}, \end{aligned}$$

which is not equal to $R = e(g,g)^{s_A t}$. Subsequently, $\mathcal{A}$ computes $R'^{\frac{1}{s_A}}$, which can only be considered as a random value. In addition, if there are two trapdoors $t_{0,i}$ and $t_{0,j}$ in DB_0, with the different random values r_i and r_j, $\mathcal{C}$ returns two different ciphertexts T_i and T_j, which $\mathcal{A}$ cannot distinguish. Then, $\mathcal{A}$ can only select $b' = 0$ or $b' = 1$ at random. Therefore,

$$\mathsf{Adv}_{\mathcal{A}}^{IND-CPA}(\lambda) = |Pr[b' = b|\mathcal{A}(DB_0, DB_1, T)] - \frac{1}{2}| \le \epsilon,$$

where ϵ is negligible. In addition, without the master secret key s, $\mathcal{A}$ cannot forge other secret keys. Therefore, Theorem 4 is proven. □

Theorem 5. *Our proxy re-encryption approach for the* **Match** *phase is secure under the chosen-plaintext attack model.*

Proof. Similar to Theorem 4, we can establish the security of the proposed proxy re-encryption technique under the chosen-plaintext attack model. Without the edit re-encryption key $s_{A,0}$, $\mathcal{A}$ cannot obtain any information from the ciphertexts or the match results. Therefore, the detailed proof can be omitted. □

Theorem 6. *If Theorems 4 and 5 are proven, for any adversary, the trapdoor, policy, identity, and match privacy will not be disclosed.*

Proof. In our work, the knowledge owner cannot receive any response or data from other entities, and the knowledge editor can only receive the match results from the service nodes. Since the security of our encryption and proxy re-encryption approaches is demonstrated under the chosen-plaintext attack model, the owner and editor cannot obtain other trapdoors with their secret keys but the edit privilege. Since Theorem 5 is proven, the service nodes cannot obtain the trapdoor with the match ciphertext, match key, re-encryption key, and match results. Then, nobody can break the trapdoor privacy. Without the master secret key s and the edit re-encryption key s_A, the service nodes cannot forge secret keys from the received match keys or infer the editor $\mathcal{A}$'s identity. Moreover, without the re-encryption key $s_{A,0}$, the editor $\mathcal{A}$ also cannot forge secret keys from his or her own secret key. Then, the identity privacy is preserved. For the editor or the other entities, it is difficult to infer policy content or the exact policy size from the ciphertext,

which preserves policy privacy. In addition, in PIRB-II, it is also difficult to distinguish which identity is unsuitable for the policy as the service nodes must execute the complete **Match** phase to obtain the match results, which preserves the match privacy. Therefore, Theorem 6 is proven. □

Theorem 7. *Any editor with malicious behaviors can be traced in PIRB.*

Proof. An editor can engage in malicious behaviors in the **Match** and **Edit** phases. In the **Match** phase, the editor $\mathcal{A}$ can generate the match key without the random value s_A as

$$K' = (K'_N, K'_{N-1}, \cdots, K'_1, K'_0),$$
$$K'_l = g^{s_{A,0} s H_1(ID_A)^l}, 1 \le l \le N,$$
$$K'_0 = g,$$

to break the trapdoor privacy. Specifically, the service nodes can compute

$$R' = \prod_{l=1}^{N} e\left(T_{j,2,l}, K'^{\frac{1}{s_{A,0}}}_{k,l}\right) \cdot e\left(T_{j,2,0}, K'_{k,0}\right)$$
$$= e(g,g)^{\left(\sum_{l=1}^{N} b_l H_1(ID_A)^l + b_0\right)} = e(g,g)^t,$$

to directly obtain the trapdoor. However, since Theorem 6 is proven, $\mathcal{A}$ cannot forge secret keys from his or her own secret key. Then, to execute the *Match* phase, $\mathcal{A}$ must generate the match key with his or her own secret key and provide the virtual identity VID_{u_A} for the associated re-encryption key $s_{A,0}$. With VID_{u_A}, the consortium nodes can trace the editor $mathcal A$. Similarly, if the editor $\mathcal{A}$ displays illegal edit behaviors, $\mathcal{A}$ can still be traced with the virtual identity VID_{u_A}. Therefore, Theorem 7 is proven. □

5.3. Complexity Analysis

In the complexity analysis, we analyze the computational complexity and communication complexity among the consortium blockchain schemes supporting one policy for a batch of users, i.e., DerlerNDSS19, XuTIFS23, PIRB-I and PIRB-II. The complexity analysis results are shown in Table 3.

Table 3. Theoretical computational complexity and communication complexity.

Scheme	Entity	Computational Complexity	Communication Complexity
DerlerNDSS19	Knowledge Owner	$\mathcal{O}(N_P)T_e + \mathcal{O}(1)T_p + \mathcal{O}(N_P)T_h + \mathcal{O}(1)T_{SE}$	$\mathcal{O}(N_P)\|1^\lambda\|$
	Service Node	—	$(\phi_o + \phi_s\phi_e)\mathcal{O}(N_P)\|1^\lambda\|$
	Knowledge Editor	$\phi_s(\mathcal{O}(N_P)T_e + \mathcal{O}(1)T_p + \mathcal{O}(N_P)T_h + \mathcal{O}(1)T_{SD})$	$\phi_s\mathcal{O}(N_P)\|1^\lambda\|$
XuTIFS23	Knowledge Owner	$\mathcal{O}(N_P)T_e + \mathcal{O}(1)T_p + \mathcal{O}(N_P)T_h + \mathcal{O}(1)T_{SE}$	$\mathcal{O}(N_P)\|1^\lambda\|$
	Service Node	—	$(\phi_o + \phi_s\phi_e)\mathcal{O}(N_P)\|1^\lambda\|$
	Knowledge Editor	$\phi_s(\mathcal{O}(N_P)T_e + \mathcal{O}(1)T_p + \mathcal{O}(N_P)T_h + \mathcal{O}(1)T_{SD})$	$\phi_s\mathcal{O}(N_P)\|1^\lambda\|$
PIRB-I	Knowledge Owner	$\mathcal{O}(N)T_e + \mathcal{O}(1)T_p + \mathcal{O}(N)T_h$	$\mathcal{O}(N)\|1^\lambda\|$
	Service Node	$\phi_o\phi_e(\mathcal{O}(N)T_e + \mathcal{O}(N)T_p)$	$(\phi_o + \phi_e)\mathcal{O}(N)\|1^\lambda\| + \phi_s\phi_e\mathcal{O}(1)\|1^\lambda\|$
	Knowledge Editor	$\mathcal{O}(N)T_e + \phi_s(\mathcal{O}(1)T_e + \mathcal{O}(1)T_h)$	$\mathcal{O}(N)\|1^\lambda\| + \phi_s\mathcal{O}(1)\|1^\lambda\|$
PIRB-II	Knowledge Owner	$1.5\mathcal{O}(N)T_e + \mathcal{O}(1)T_p + \mathcal{O}(N)T_h$	$1.5\mathcal{O}(N)\|1^\lambda\|$
	Service Node	$2\phi_o\phi_e(\mathcal{O}(N)T_e + \mathcal{O}(N)T_p)$	$2(\phi_o + \phi_e)\mathcal{O}(N)\|1^\lambda\| + \phi'_s\phi_e\mathcal{O}(1)\|1^\lambda\|$
	Knowledge Editor	$2\mathcal{O}(N)T_e + \phi_s(\mathcal{O}(1)T_e + \mathcal{O}(1)T_h)$	$2\mathcal{O}(N)\|1^\lambda\| + \phi'_s\mathcal{O}(1)\|1^\lambda\|$

Notation Introduction. In Table 3, N is the preset maximum policy size in PIRB, and N_P denotes the size of the policy. T_p, T_h, T_e, T_{SE} and T_{SD} denote the time for executing an operation of pairing, hash, exponent, symmetric encryption, and symmetric decryption, re-

spectively. ϕ_o, ϕ_s and ϕ_e denote the number of total knowledge owners, suitable knowledge owners and knowledge editors, respectively. Note that the number of knowledge editors is the number of required knowledge editors for the match and edit service per unit of time. In PIRB-II, the number of suitable knowledge owners is actually lower compared to other schemes, denoted as ϕ_s'. Moreover, the constant coefficients in the complexity analysis of PIRB-II serve the purpose of facilitating a comparative evaluation with PIRB-I.

Complexity Analysis Results. From Table 3, on the knowledge owner side, the computational and communication complexity in PIRB are, respectively, $\mathcal{O}(N)T_e + \mathcal{O}(1)T_p + \mathcal{O}(N)T_h$ and $\mathcal{O}(N)|1^\lambda|$, which is similar to other schemes. On the service node side, the computational and communication complexity in PIRB are, respectively, $\phi_o\phi_e(\mathcal{O}(N)T_e + \mathcal{O}(N)T_p)$ and $(\phi_o + \phi_e)\mathcal{O}(N)|1^\lambda| + \phi_s\phi_e\mathcal{O}(1)|1^\lambda|$, which is slightly higher than other schemes. This is because, in PIRB, to find suitable owners while hiding the policies, we designed the **Match** phase, and the service nodes perform the phase to decrease the overheads of the knowledge editors. On the knowledge editor side, the computational and communication complexity in PIRB are, respectively, $\mathcal{O}(N)T_e + \phi_s(\mathcal{O}(1)T_e + \mathcal{O}(1)T_h)$ and $\mathcal{O}(N)|1^\lambda| + \phi_s\mathcal{O}(1)|1^\lambda|$, which is much lower than DerlerNDSS19 and XuTIFS23. This is because the service nodes bear the overheads of pairing to reduce the computational complexity and perform the **Match** phase to prevent the editors from searching for suitable owners to reduce the communication complexity. Moreover, because the exact number of exponent operations in PIRB is much lower than in DerlerNDSS19 and XuTIFS23, the total overhead in PIRB is also lower. PIRB-I and PIRB-II involve a trade-off between computational and communication complexity.

6. Performance Evaluation

In this section, based on an original model deployed on the FISCO blockchain, we evaluate the performance of our schemes, i.e., PIRB-I and PIRB-II, compared with two recent redactable blockchain schemes, DerlerNDSS19 [18] and XuTIFS23 [19].

6.1. Experimental Configuration

We deploy the FISCO blockchain on a Ubuntu-18.04.6 virtual machine created on the VMware Workstation 16 Pro, with 4 CPUs with 4GB RAM and 20 GB SCSI. Our implementation and execution of the prototype are conducted on a laptop with 64-bit CPU, Windows 10, Intel(R) Core(TM) i7-10510U 1.80 GHz with 16 GB of RAM. In DerlerNDSS19, XuTIFS23 and our own works, we employ the Java programming language and utilize the JPBC library to realize the computational function, choosing the Type A curve with 80-bit security. Note that since PIRB does not rely on a specific curve, the experimental results can be extrapolated to curves with elevated security parameters, despite the trade-off between the security level and efficiency. To ensure statistical robustness, each experiment is executed ten times, and the resultant average running time is computed as the definitive experimental outcome. The policy size, which is a key parameter in DerlerNDSS19 and XuTIFS23, ranges from 10 to 100, aligning with the preset maximum policy size in our constructions.

6.2. Experimental Evaluation

We focus on the computational overheads of the **Hash**, **Match** and **Edit** phases.

On the knowledge owner side. We evaluate the computational and communication costs on the worker side and illustrate the experimental results in Figure 4. Particularly, because the computational and communication overhead is only correlated with the policy size in DerlerNDSS19, XuTIFS23 and the preset maximum policy size in our constructions, we consider the policy size from 10 to 100. The security parameter λ is set to 80. Figure 4a shows that PIRB-I and PIRB-II are more efficient than DerlerNDSS19 and XuTIFS23 regarding the computational costs. This is because an owner executes at least $12N_P$ hash and $12N_P$ exponent operations in DerlerNDSS19 and XuTIFS23; however, an owner executes only N hash and $2N$ exponent operations in PIRB-I and N hash and $3N$ exponent operations in

PIRB-II. In addition, Figure 4a shows that the computational overheads of all four schemes grow linearly with the size of the policies. Figure 4b shows that the four schemes have similar communication costs.

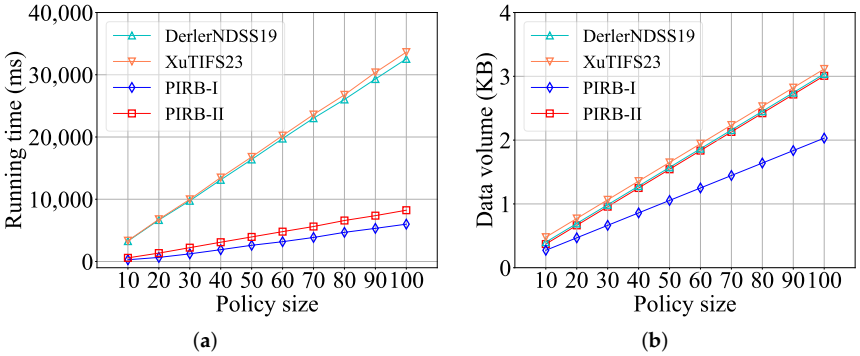

Figure 4. Computational and communication costs with respect to policy size on the owner side. (**a**) Computational costs with respect to policy size on the owner side. (**b**) Communication costs with respect to policy size on the owner side.

On the service nodes. Next, we evaluate the computational costs on the service nodes for an editor and illustrate the experimental results in Figure 5. Specifically, we consider the number of owners from 10 to 100 and set the preset maximum policy size as 10, 20, 50 and 100. As Figure 5 shows, the service nodes execute $\phi_o N + \phi_s N$ and $2\phi_o N + 2\phi'_s N$ exponent and pairing operations in PIRB-I and PIRB-II, respectively, where $\phi'_s < \phi_s$ actually.

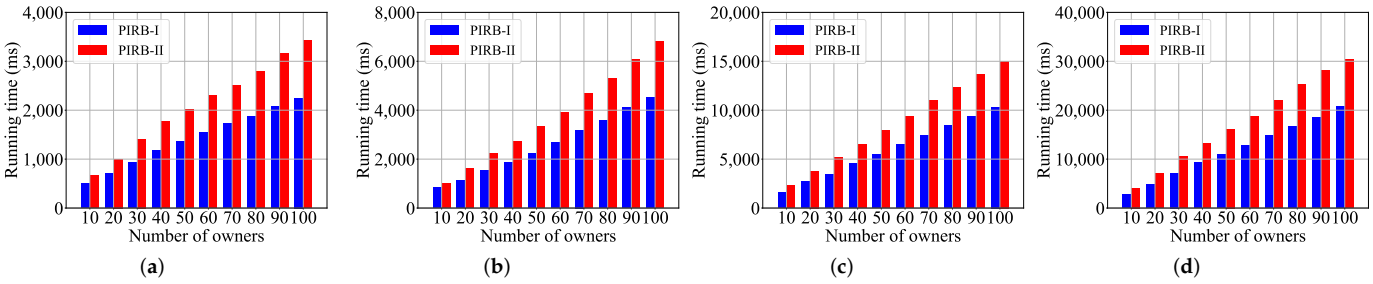

Figure 5. Computational costs with respect to number of owners on the edge devices. (**a**) Computational costs with respect to number of owners with policy size 10. (**b**) Computational costs with respect to number of owners with policy size 20. (**c**) Computational costs with respect to number of workers with policy size 50. (**d**) Computational costs with respect to number of workers with policy size 100.

On the knowledge editor side. Finally, we evaluate the computational and communication costs on the editor side and illustrate the experimental results in Figures 6 and 7. Particularly, because the computational and communication overhead is correlated with the policy size, the suitable owner number in DerlerNDSS19 and XuTIFS23 and the preset maximum policy size, the suitable owner number, in our constructions, we consider the policy size from 10 to 100 and set the number of suitable owners as 10, 20, 50 and 100. Notably, the number of suitable owners in PIRB-II is actually less than in other schemes. Therefore, we set the number of suitable owners as 5, 10, 25 and 50. The security parameter λ is set to 80. Figure 6 shows that PIRB-I and PIRB-II have better computational performance than DerlerNDSS19 and XuTIFS23. This is because an editor executes at least $12\phi_s N_P$ hash and $12\phi_s N_P$ exponent operations in DerlerNDSS19 and XuTIFS23; however, an owner executes only ϕ_s hash and $N + \phi_s$ exponent operations in PIRB-I and ϕ_s hash and $2N + \phi_s$ exponent operations in PIRB-II. Figure 7 shows that PIRB-I and PIRB-II have lower communication costs than DerlerNDSS19 and XuTIFS23. This is because, in DerlerNDSS19 and XuTIFS23, the communication costs are $80\phi_s(3N_P + 2)$ and $80\phi_s(3N_P + 4)$, respectively, which brings

overheads proportional to the number of suitable owners ϕ_s. However, in PIRB-I and PIRB-II, with the **Match** phase to decrease the communication costs, the communication costs are $80(N + 4\phi_s)$ and $80(2N + 4\phi'_s)$, respectively, where $\phi'_s < \phi_s$ actually. Notably, Figures 6 and 7 show that PIRB-I and PIRB-II involve a trade-off between computational and communication overheads. Specifically, while PIRB-I demonstrates superior computational efficiency compared to PIRB-II, PIRB-II exhibits a distinct advantage in terms of communication overhead. This is due to the growing number of suitable owners, leading to a situation where the advantage of PIRB-I's uncomplicated policy format is outweighed by the editors' policy functionality in PIRB-II, which effectively minimizes the number of returned match results.

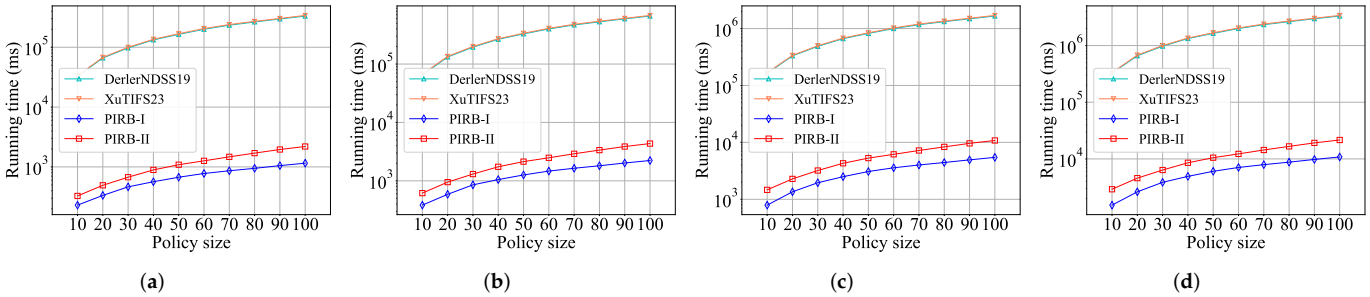

Figure 6. Computational costs with respect to policy size on the editor side. (**a**) Computational costs with respect to policy size with 10 suitable owners. (**b**) Computational costs with respect to policy size with 20 suitable owners. (**c**) Computational costs with respect to policy size with 50 suitable owners. (**d**) Computational costs with respect to policy size with 100 suitable owners.

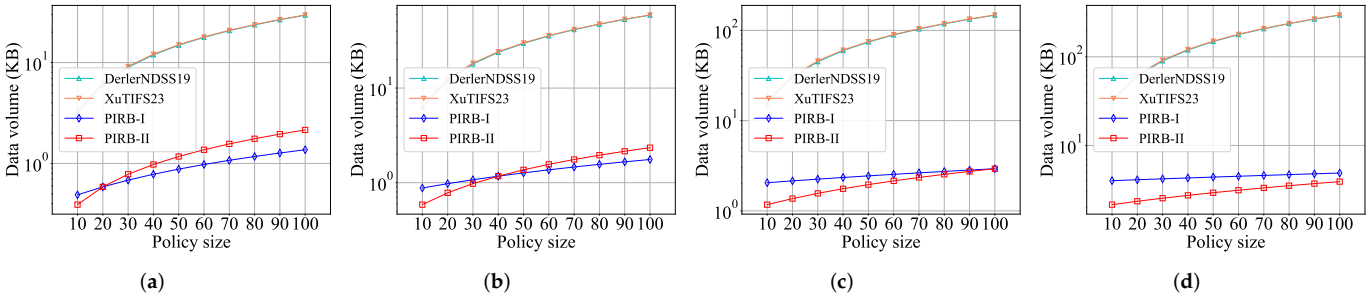

Figure 7. Communication costs with respect to policy size on the editor side. (**a**) Communication costs with respect to policy size with 10 suitable owners. (**b**) Communication costs with respect to policy size with 20 suitable owners. (**c**) Communication costs with respect to policy size with 50 suitable owners. (**d**) Communication costs with respect to policy size with 100 suitable owners.

7. Related Works

This section introduces the related works on redactable blockchain and identity-based encryption.

In recent years, several redactable blockchain schemes have been proposed. For identity-based scenarios, Ateniese et al. [36] introduced the pioneering concept of an identity-based chameleon hash. However, this scheme exhibits susceptibility to key exposure unless an identity is never reused. Addressing this concern, Chen et al. [16] presented an identity-based chameleon hash scheme without key exposure in the random oracle model, and Bao et al. [37] subsequently proposed a hierarchical extension. Seeking to enhance the security, Xie et al. [30] devised an identity-based chameleon hash scheme within the standard model. Expanding upon this, Li et al. [31] formulated an efficient identity-based chameleon hash scheme to optimize the size of public parameters and reduce computational complexity. Building upon this groundwork [16], Zhou et al. [17] introduced an identity-based fine-grained redactable blockchain. However, the aforemen-

tioned identity-based schemes are limited in their capacity to support flexible policies and to implement bilateral access control.

Hence, we then focus on policy-based redactable blockchain schemes that offer the capacity for flexible policies. Ateniese et al. [24] introduced the pioneering redactable blockchain leveraging chameleon hash techniques, securing the trapdoor through public key infrastructure for trapdoor access control. However, this scheme suffers from limitations such as inflexible privilege management due to a single fixed public key, compromising modifier identity privacy and susceptibility to collision vulnerabilities. Moreover, the scheme grants the modifier excessive rewriting power at the block level, failing to accommodate finer-grained transaction-level needs. Seeking enhanced security and flexibility, Derler et al. [18] devised the policy-based chameleon hash (PCH) framework, integrating chameleon hash with ephemeral trapdoors (CHET) and attribute-based encryption (ABE). CHET introduces a safeguard against collisions by combining long-term and ephemeral trapdoors, while ABE imparts flexibility and privacy preservation through the linear secret-sharing scheme. Subsequently, Tian et al. [26] extended PCH with black-box accountability, yet this introduced inadvertent susceptibility to semi-trusted modifiers due to key leakage. To bolster accountability, Xu et al. [19] introduced identity-based signatures with existential unforgeability and proposed an attribute-based traitor tracing scheme. Moreover, Xu et al. [27] introduced number resistance and expiration dates to govern rewriting privileges, enforced through double-authentication preventing signatures, supplemented by a monetary penalty system to deter malicious behavior. For permissionless settings, Deuber et al. [25] devised an efficient redactable blockchain employing consensus-based voting. Notably, this solution entails traversing transaction logs for redaction validation and voting, incurring time costs. In decentralized settings, Jia et al. [28] proposed a decentralized chameleon hash for redactable blockchains, collaboratively generating keys across untrusted blockchain nodes. Ma et al. [29] introduced a decentralized policy-based chameleon hash employing multi-authority attribute-based encryption to manage editing privileges. However, the two approaches in [28,29] neglect data and policy privacy and incur substantial communication overheads for intricate policies. Regrettably, none of the above schemes preserves policy privacy. Guo et al. [38] proposed a decentralized policy-hidden fine-grained redactable blockchain. Moreover, in cases involving bilateral access control scenarios, none of these schemes offer viable solutions.

The encryption of trapdoors in the CHET framework necessitates the use of an identity-based encryption scheme. The pioneering work by Boneh et al. [35] introduced the first identity-based encryption scheme utilizing the Weil pairing, achieving chosen-ciphertext security within the random oracle model. Subsequent efforts by Boneh et al. [39,40] yielded two distinct schemes, enhancing chosen-ciphertext security within the selective identity model and the full identity model, respectively. Despite these advancements, concerns regarding efficiency persist. Waters et al. [41] proposed an efficient identity-based encryption scheme, ensuring chosen-ciphertext security in the full identity model, albeit relying on numerous public parameters. To mitigate this parameter proliferation, Gentry et al. [42] presented a practical identity-based encryption scheme. Moreover, efforts to augment the capabilities of identity-based encryption led to the conception of tightly secure identity-based encryption [43] and function-private identity-based encryption [44]. However, a notable limitation among the aforementioned schemes is their inability to achieve one policy for a batch of users. This gap was addressed by Sun et al. [45] through the introduction of efficient matchmaking encryption, leveraging polynomial functions to facilitate flexible data sharing. Nevertheless, for editors, the absence of an effective matching mechanism [6,46] introduces the potential for superfluous communication overheads in the endeavor to select appropriate ciphertexts.

8. Conclusions

In this paper, we have introduced a novel privacy-preserving identity-based redactable blockchain (PIRB), which leverages the combined techniques of identity-based encryption

and chameleon hash with ephemeral trapdoors. The PIRB scheme specifically addresses the need for redactable blockchains in identity-based scenarios, ensuring the preservation of trapdoor, identity, policy, and match privacy. Notably, PIRB employs a polynomial function technique to achieve one identity-based policy for a batch of users, offering both one-way (PIRB-I) and bilateral (PIRB-II) access control paradigms. Moreover, accountability is upheld through the utilization of virtual identities. Our constructions have been rigorously validated through correctness analysis, affirming the successful execution of match and edit operations. The security analysis under a chosen-plaintext attack attests to the robust preservation of trapdoor, identity, policy, and match privacy, while also ensuring accountability. The detailed complexity analysis, along with empirical evidence from a prototype implementation, underscores the tangible efficiency enhancements that our constructions bring to practical deployment. As a direction for future exploration, we envision augmenting our constructions with fine-grained privilege management encompassing read, use, and edit privileges, thereby enhancing the adaptability of blockchain technology across emerging applications.

Author Contributions: Conceptualization, Y.X.; Formal analysis, Z.L.; Investigation, Y.X.; Methodology, Y.X.; Software, Z.L.; Supervision, Z.L.; Writing—original draft, Y.X.; Writing—review and editing, Z.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the "National Key R&D Program of China" (Grant Nos. 2021YFB2700500 and 2021YFB2700503), the National Natural Science Foundation of China (Grant No. 62232002), and the Shandong Provincial Key Research and Development Program (Grant No. 2021CXGC010106).

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

References

1. Zhou, B.; Li, H.; Xu, L. An Authentication Scheme Using Identity-based Encryption & Blockchain. In Proceedings of the 2018 IEEE Symposium on Computers and Communications (ISCC 2018), Natal, Brazil, 25–28 June 2018; pp. 556–561.
2. Babu, E.S.; Dadi, A.K.; Singh, K.K.; Nayak, S.R.; Bhoi, A.K.; Singh, A. A distributed identity-based authentication scheme for internet of things devices using permissioned blockchain system. *Expert Syst. J. Knowl. Eng.* **2022**, *39*, e12941. [CrossRef]
3. Huang, C.; Wang, W.; Liu, D.; Lu, R.; Shen, X. Blockchain-Assisted Personalized Car Insurance with Privacy Preservation and Fraud Resistance. *IEEE Trans. Veh. Technol.* **2023**, *72*, 3777–3792. [CrossRef]
4. Xue, J.; Xu, C.; Zhao, J.; Ma, J. Identity-based public auditing for cloud storage systems against malicious auditors via blockchain. *Sci. China Inf. Sci.* **2019**, *62*, 32104:1–32104:16. [CrossRef]
5. Yuan, Y.; Zhang, J.; Xu, W.; Li, Z. Identity-based public data integrity verification scheme in cloud storage system via blockchain. *J. Supercomput.* **2022**, *78*, 8509–8530. [CrossRef]
6. Li, Y.; Zhao, M.; Li, Z.; Zhang, W.; Dong, J.; Wu, T.; Zhang, C.; Zhu, L. Achieving a Blockchain-based Privacy-preserving Quality-aware Knowledge Marketplace in Crowdsensing. In Proceedings of the 20th IEEE International Conference on Embedded and Ubiquitous Computing (EUC 2022), Wuhan, China, 9–11 December 2022; pp. 90–97.
7. Ren, H.; Li, H.; Liu, D.; Xu, G.; Cheng, N.; Shen, X. Privacy-Preserving Efficient Verifiable Deep Packet Inspection for Cloud-Assisted Middlebox. *IEEE Trans. Cloud Comput.* **2022**, *10*, 1052–1064. [CrossRef]
8. Sharma, P.; Moparthi, N.R.; Namasudra, S.; Shanmuganathan, V.; Hsu, C. Blockchain-based IoT architecture to secure healthcare system using identity-based encryption. *Expert Syst. J. Knowl. Eng.* **2022**, *39*, e12915. [CrossRef]
9. Ren, H.; Xu, G.; Qi, H.; Zhang, T. PriFR: Privacy-preserving Large-scale File Retrieval System via Blockchain for Encrypted Cloud Data. In Proceedings of the 2023 IEEE 9th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing (HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS), New York, NY, USA, 6–8 May 2023; pp. 16–23.
10. 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]
11. 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**, *20*, 4245–4257. [CrossRef]
12. Huang, C.; Liu, D.; Yang, A.; Lu, R.; Shen, X. Multi-client Secure and Efficient DPF-based Keyword Search for Cloud Storage. *IEEE Trans. Dependable Secur. Comput.* **2023**, 1–18. [CrossRef]
13. Regulation, P. General data protection regulation. *Intouch* **2018**, *25*, 1–5.
14. Wu, W.; Li, M.; Qu, K.; Zhou, C.; Shen, X.; Zhuang, W.; Li, X.; Shi, W. Split Learning Over Wireless Networks: Parallel Design and Resource Management. *IEEE J. Sel. Areas Commun.* **2023**, *41*, 1051–1066. [CrossRef]

15. 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. [CrossRef]
16. Chen, X.; Zhang, F.; Susilo, W.; Tian, H.; Li, J.; Kim, K. Identity-Based Chameleon Hash Scheme without Key Exposure. In Proceedings of the Information Security and Privacy—15th Australasian Conference (ACISP 2010), Sydney, Australia, 5–7 July 2010; Lecture Notes in Computer Science; Steinfeld, R., Hawkes, P., Eds.; Springer: Berlin/Heidelberg, Germany, 2010; Volume 6168, pp. 200–215.
17. Zhou, G.; Ding, X.; Han, H.; Zhu, A. Fine-Grained Redactable Blockchain Using Trapdoor-Hash. *IEEE Internet Things J.* **2023**, 1. [CrossRef]
18. Derler, D.; Samelin, K.; Slamanig, D.; Striecks, C. Fine-Grained and Controlled Rewriting in Blockchains: Chameleon-Hashing Gone Attribute-Based. In Proceedings of the 26th Annual Network and Distributed System Security Symposium (NDSS 2019), San Diego, CA, USA, 24–27 February 2019; The Internet Society: Reston, VA, USA, 2019.
19. Xu, S.; Huang, X.; Yuan, J.; Li, Y.; Deng, R.H. Accountable and Fine-Grained Controllable Rewriting in Blockchains. *IEEE Trans. Inf. Forensics Secur.* **2023**, *18*, 101–116. [CrossRef]
20. Ren, H.; Li, H.; Liu, D.; Xu, G.; Shen, X.S. Enabling Secure and Versatile Packet Inspection with Probable Cause Privacy for Outsourced Middlebox. *IEEE Trans. Cloud Comput.* **2022**, *10*, 2580–2594. [CrossRef]
21. Zhang, C.; Zhao, M.; Wu, T.; Zhang, W.; Fan, Q.; Zhu, L. Towards Secure Bilateral Friend Query with Conjunctive Policy Matching in Social Networks. In Proceedings of the IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom 2022), Melbourne, Australia, 17–19 December 2022; pp. 98–105.
22. 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]
23. Wu, W.; Chen, N.; Zhou, C.; Li, M.; Shen, X.; Zhuang, W.; Li, X. Dynamic RAN Slicing for Service-Oriented Vehicular Networks via Constrained Learning. *IEEE J. Sel. Areas Commun.* **2021**, *39*, 2076–2089. [CrossRef]
24. Ateniese, G.; Magri, B.; Venturi, D.; Andrade, E.R. Redactable Blockchain—or—Rewriting History in Bitcoin and Friends. In Proceedings of the 2017 IEEE European Symposium on Security and Privacy (EuroS&P 2017), Paris, France, 26–28 April 2017; pp. 111–126.
25. 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 2019), San Francisco, CA, USA, 19–23 May 2019; pp. 124–138.
26. 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 ACSAC '20: Annual Computer Security Applications Conference, Austin, TX, USA, 7–11 December 2020; ACM: New York, NY, USA, 2020; pp. 813–828.
27. 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]
28. 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]
29. 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]
30. Xie, Z.; Shen, Q.; Li, C.; Dong, J.; Fang, Y. Identity-Based Chameleon Hash without Random Oracles and Application in the Mobile Internet. In Proceedings of the ICC 2021—IEEE International Conference on Communications, Montreal, QC, Canada, 14–23 June 2021; pp. 1–6.
31. Li, C.; Shen, Q.; Xie, Z.; Dong, J.; Fang, Y.; Wu, Z. Efficient Identity-Based Chameleon Hash for Mobile Devices. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2022), Singapore, 23–27 May 2022; pp. 3039–3043.
32. Krawczyk, H.; Rabin, T. Chameleon Signatures. In Proceedings of the Network and Distributed System Security Symposium (NDSS 2000), San Diego, CA, USA, 3–4 February 2000; The Internet Society: Reston, VA, USA, 2000.
33. 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; Lecture Notes in Computer Science; Fehr, S., Ed.; Springer: Berlin/Heidelberg, Germany, 2017; Volume 10175, pp. 152–182.
34. Samelin, K.; Slamanig, D. Policy-Based Sanitizable Signatures. In Proceedings of the Topics in Cryptology—CT-RSA 2020—The Cryptographers' Track at the RSA Conference 2020, San Francisco, CA, USA, 24–28 February 2020; Lecture Notes in Computer Science; Jarecki, S., Ed.; Springer: Berlin/Heidelberg, Germany, 2020; Volume 12006, pp. 538–563.
35. Boneh, D.; Franklin, M.K. Identity-Based Encryption from the Weil Pairing. In Proceedings of the Advances in Cryptology—CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, CA, USA, 19–23 August 2001; Lecture Notes in Computer Science; Kilian, J., Ed.; Springer: Berlin/Heidelberg, Germany, 2001; Volume 2139, pp. 213–229.
36. Ateniese, G.; de Medeiros, B. Identity-Based Chameleon Hash and Applications. In Proceedings of the Financial Cryptography, 8th International Conference (FC 2004), Key West, FL, USA, 9–12 February 2004; Revised Papers; Lecture Notes in Computer Science; Juels, A., Ed.; Springer: Berlin/Heidelberg, Germany, 2004; Volume 3110, pp. 164–180.

37. Bao, F.; Deng, R.H.; Ding, X.; Lai, J.; Zhao, Y. Hierarchical Identity-Based Chameleon Hash and Its Applications. In Proceedings of the Applied Cryptography and Network Security—9th International Conference, ACNS 2011, Nerja, Spain, 7–10 June 2011; Lecture Notes in Computer Science; López, J., Tsudik, G., Eds.; Volume 6715, pp. 201–219.
38. Guo, H.; Tao, X.; Zhao, M.; Wu, T.; Zhang, C.; Xue, J.; Zhu, L. Decentralized Policy-Hidden Fine-Grained Redaction in Blockchain-Based IoT Systems. *Sensors* **2023**, *23*, 7105. [CrossRef] [PubMed]
39. Boneh, D.; Boyen, X. Efficient Selective-ID Secure Identity-Based Encryption without Random Oracles. In Proceedings of the Advances in Cryptology—EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, 2–6 May 2004; Lecture Notes in Computer Science; Cachin, C., Camenisch, J., Eds.; Springer: Berlin/Heidelberg, Germany, 2004; Volume 3027, pp. 223–238.
40. Boneh, D.; Boyen, X. Secure Identity Based Encryption without Random Oracles. In Proceedings of the Advances in Cryptology—CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, CA, USA, 15–19 August 2004; Lecture Notes in Computer Science; Franklin, M.K., Ed.; Springer: Berlin/Heidelberg, Germany, 2004; Volume 3152, pp. 443–459.
41. Waters, B. Efficient Identity-Based Encryption without Random Oracles. In Proceedings of the Advances in Cryptology—EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, 22–26 May 2005; Lecture Notes in Computer Science; Cramer, R., Ed.; Springer: Berlin/Heidelberg, Germany, 2005; Volume 3494, pp. 114–127.
42. Gentry, C. Practical Identity-Based Encryption without Random Oracles. In Proceedings of the Advances in Cryptology—EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, 28 May–1 June 2006; Lecture Notes in Computer Science; Vaudenay, S., Ed.; Volume 4004, pp. 445–464.
43. Chen, J.; Wee, H. Fully, (Almost) Tightly Secure IBE and Dual System Groups. In Proceedings of the Advances in Cryptology—CRYPTO 2013—33rd Annual Cryptology Conference, Santa Barbara, CA, USA, 18–22 August 2013; Proceedings, Part II; Lecture Notes in Computer Science; Canetti, R., Garay, J.A., Eds.; Springer: Berlin/Heidelberg, Germany, 2013; Volume 8043, pp. 435–460.
44. Boneh, D.; Raghunathan, A.; Segev, G. Function-Private Identity-Based Encryption: Hiding the Function in Functional Encryption. In Proceedings of the Advances in Cryptology—CRYPTO 2013—33rd Annual Cryptology Conference, Santa Barbara, CA, USA, 18–22 August 2013; Proceedings, Part II; Lecture Notes in Computer Science; Canetti, R., Garay, J.A., Eds.; Springer: Berlin/Heidelberg, Germany, 2013; Volume 8043, pp. 461–478.
45. Sun, J.; Xu, G.; Zhang, T.; Yang, X.; Alazab, M.; Deng, R.H. Privacy-Aware and Security-Enhanced Efficient Matchmaking Encryption. *IEEE Trans. Inf. Forensics Secur.* **2023**, *18*, 4345–4360. [CrossRef]
46. 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]