This article has been accepted for publication in IEEE Transactions on Dependable and Secure Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2024.3386690

Efficient Fine-Grained Data Sharing based on Proxy Re-encryption in IIoT

Qingyang Zhang, Yujie Fu, Jie Cui, Debiao He, Hong Zhong

Abstract—With the development of the industrial Internet of Things (IIoT), the amount of data generated by industrial manufacturing equipment will increase. To reduce the cost of data management while achieving secure data sharing, data owners generally upload the resulting ciphertexts to a cloud server after encrypting their data. Attribute-based encryption (ABE) is a valuable technology that implements fine-grained access control over shared information; however, its computational complexity is not suitable for resource-constrained IIoT devices, making it difficult to apply directly to an IIoT environment. To address this problem, we design a fine-grained data sharing scheme based on proxy re-encryption in IIoT. In the proposed scheme, data files are encrypted through an identity-based encryption and a data owner can authorize a semi-trusted proxy server to transform the ciphertext into an ABE ciphertext. This realizes fine-grained access control and decreases a data owner's computational cost in data sharing. In addition, the computational burden is outsourced to a cloud server, and users only need to perform simple computing operations. A formal security proof indicates the proposed scheme's selective chosen-plaintext attack security. Theoretical and experimental analyses illustrate that our construction is more efficient than previous schemes.

Index Terms—Industrial internet of things, data sharing, attribute-based encryption, proxy re-encryption, access control

1 Introduction

With the rapid progress of wireless communication technology, the Internet of Things (IoT) [1] has become a promising paradigm that significantly promotes the connection between the Internet and various physical objects in the real world. In the last few years, IoT has been employed in various fields, particularly in the industrial field. Industrial IoT (IIoT) [2], [3] upgrades traditional industries to intelligent industries, which improves manufacturing efficiency and reduces production costs. The IIoT needs real-time online processing to improve the efficiency and cost-effectiveness of industrial production. Therefore, encryption and other confidential transmissions should be lightweight [4]. However, with a substantial increase in IIoT data, the processing and sharing of massive amounts of real-time data have become the main challenge [5]. Cloud computing [6], [7] is leveraged to handle large amounts of IIoT data. However, the cloud server is "honest-but-curious", which means it performs its tasks honestly but does not ensure the privacy of user data. In this case, to achieve secure data sharing, data owners generally adopt an encryption mechanism [8] to upload encrypted data to a cloud server. As a novel cryptographic solution, attribute-based encryption (ABE) [9] is usually used to realize one-to-many data sharing mode and implement fine-grained access control for encrypted data [10], [11], [12]. Fig. 1 shows the access control model of ABE in the IIoT. IIoT devices are deployed by the data owner to collect real-time data, which is encrypted based on ABE with the specific access policy. And, the generated ciphertext can be decrypted if the data consumer's attributes match the access policy.

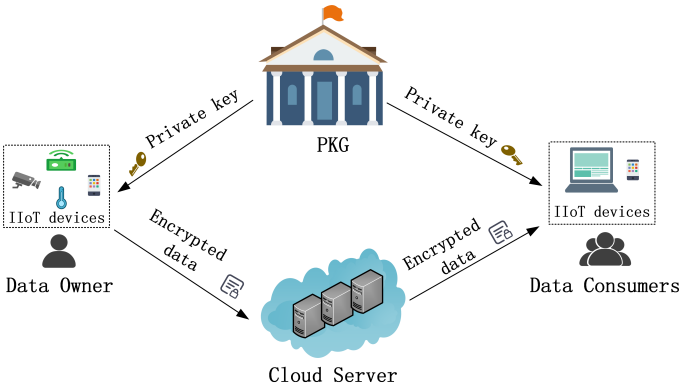


Fig. 1: Access Control Model of ABE in the IIoT

However, ABE has a high computational cost for encryption and decryption, especially for typical IIoT devices, which have resource-constrained in terms of computing and storage capacity. Taking time consumption on essential cryptographic pairing operation for ABE as an example, it is almost 600 ms under 128-bit security level on Raspberry Pi 3b with 1.2GHz ARM Cortex-A53 CPU, while an IIoT device is typical without such a powerful CPU. In addition, the number of pairing operations for ABE encryption and decryption increases linearly with the number of attributes. Thus the high overhead of ABE fundamentally hinders the wide deployment in the IIoT [13].

- *Q. Zhang, Y. Fu, J. Cui and H. Zhong are with the School of Computer Science and Technology, Anhui University, Hefei 230039, China, the Anhui Engineering Laboratory of IoT Security Technologies, Anhui University, Hefei 230039, China, and the Institute of Physical Science and Information Technology, Anhui University, Hefei 230039, China (e-mail: cuijie@mail.ustc.edu.cn).*
- *D. He is with the School of Cyber Science and Engineering, Wuhan University, Wuhan 430072, China and the Shanghai Key Laboratory of PrivacyPreserving Computation, MatrixElements Technologies, Shanghai 201204, China (email: hedebiao@163.com).*

To solve the problem of high ABE encryption cost, Hohenberger and Waters [14] applied an online/offline technology to ABE. Some complex encryption computations are executed during the offline stage, and only a small number of computations are required in the online stage, which improves the efficiency of the encryption stage. To address the problem of the high cost of ABE decryption computation, some ABE schemes [15], [16], [17] with outsourcing decryption support have been proposed. This technology provides a mechanism for re-encrypting a complex ABE ciphertext into a simple ElGamal ciphertext, which reduces the computational cost of user decryption. However, the major drawback of aforementioned method is that access policy cannot be changed; that is, the storage service or data owner must decrypt and then encrypt data with new access policy for new shared users.

Proxy re-encryption (PRE) [18], [19], [20] is a potential solution to the above case. In PRE, a semi-trusted proxy server, usually the storage server in ABE schemes, is authorized to transform an encrypted message under one public key to another without exposing the underlying original data. By combining PRE and identity-based encryption (IBE), Zhang $et\ al.$ [21] proposed an identity-based PRE (IBPRE) scheme to realize data sharing in the IIoT, but this scheme can not realize efficient data sharing. If the data owner needs to share data with multiple data consumers, he needs to perform multiple ciphertext transformations. Attribute-based PRE (ABPRE) [22], [23], [24] was proposed by combining the notions of PRE and ABE, which can achieve one-to-many data sharing mode. In ABPRE, the data owner can allow a proxy server to transform an encrypted message created by original access policy into an encrypted message generated by new access policy, thereby allowing access policy changes and realizing flexible data sharing. Yu $et\ al.$ [25] designed an ABPRE scheme to realize data sharing in the IIoT. However, these schemes have some efficiency problems, and users must suffer from high computation cost. He $et\ al.$ [26] proposed a PRE scheme that transforms IBE encrypted ciphertext to ABE encrypted ciphertext, reducing the encryption computational overhead of the data owner. However, during the re-encryption key generation, their scheme requires the data owner to communicate with the data consumer, and can not realize non-interactive transformation. Besides, the data owner needs to bear complex computational overhead to generate a re-encryption key.

Considering these existing problems, the development of a flexible data sharing scheme in the IIoT that supports fine-grained access control of ciphertext data while reducing users' computational cost is challenging.

1.1 Related Work

In this part, we first introduce the application of ABE in data sharing and analyze the problems that may be encountered in its application in the IIoT environment. Second, we introduce the application of PRE technology in data sharing.

ABE-based data sharing. ABE can achieve a one-to-many data sharing pattern, while allowing a fine-grained access control for shared data. Sahai $et\ al.$ [9] first proposed the notion of ABE, and Goyal $et\ al.$ [27] divided ABE into key-policy ABE (KP-ABE) [27] and ciphertext-policy ABE (CP-ABE) [28]. ABE has been applied to IIoT to realize data sharing and storage [29], [30]. A user can recover the original data if their attributes match the access policy. Unfortunately, the high computational cost of ABE makes its direct application difficult in the IIoT environment. To lower the computational overhead in the encryption process, Guo $et\ al.$ [31] formally introduced online/offline encryption technology, which divides the encryption procedure into offline and online phase. In this technology, the offline phase performs complex computations, whereas the online phase only executes a few simple operations. Considering the computational cost, complexities of the access policy, and linear relationship between the number of attributes, Hohenberger $et\ al.$ [14] suggested applying the online/offline technology to ABE, which reduced the encryption computation cost for users; however, the decryption cost of users was still high.

To solve the problem of the high cost of ABE decryption computation, Green $et\ al.$ [15] suggested an ABE scheme, which allows for decryption outsourcing. Most decryption operations are delegated to a proxy decryption server, thereby reducing users' decryption cost. Wang $et\ al.$ [32] proposed a verifiable ABE scheme, which supports decryption outsourcing and verified the correctness of the transformed ciphertext. Zhang $et\ al.$ [33] designed a fully outsourced ABE scheme that outsources all complex computations to proxy servers. Ma $et\ al.$ [34] suggested an outsourced ABE scheme that was verifiable and exculpable to realize exculpability in ABE settings for the first time. However, these schemes cannot achieve flexible data sharing. They do not allow access policy changes, implying that only the user designated by the ABE ciphertext's access policy can decrypt the transformed ciphertext. However, in multiuser IIoT environments, the data owner may want to share the ciphertext data with users other than those originally specified. Therefore, access policy changes are necessary so that users beyond those originally designated by the access policy can also recover the original data.

PRE-based data sharing. PRE is a valuable cryptographic mechanism and enables newly specified users to access data while ensuring data privacy. Identity-based encryption (IBE) [37] is an efficient encryption technology that uses an arbitrarily identifiable string as the public key. Deng $et\ al.$ [35] suggested a hybrid PRE scheme, which transforms ABE encrypted ciphertext to IBE encrypted ciphertext, allowing access policies changes, and reducing users' decryption computation overhead. However, this scheme has efficiency problems when the data owner needs to share data with multiple data consumers. By combining the concepts of PRE and ABE, Shao $et\ al.$ [38] designed an online/offline ABPRE scheme to realize efficient data sharing and reduce the online computational cost of mobile devices through online/offline technology. However, their scheme is not effective for low-end mobile devices. Ma $et\ al.$ [36] suggested an ABPRE scheme aimed at low-end mobile devices that adopts an outsourcing technology to minimize the computing overhead of users. However, the encryption cost on the user side of this scheme linear growth as the number of attributes grows.

He $et\ al.$ [26] suggested the first IBE-ABE PRE scheme, which transforms a ciphertext in IBE format to a ciphertext in ABE format to realize cross-domain transformation

TABLE 1: The comparison between related works

Schemes	Online/Offline encryption	Outsourced decryption	PRE	One-to-many data sharing
Feng *et al.* [29]	×	×	×	√
Hohenberger *et al.* [14]	√	×	×	√
Green *et al.* [15]	×	√	×	√
Deng *et al.* [35]	×	–	√	×
Ma *et al.* [36]	√	√	√	√
He *et al.* [26]	×	×	√	√

(transform ciphertext in one encryption format into ciphertext in another encryption format). Although this is similar to the concept of our scheme, there are some significant differences between their model and ours. During the re-encryption key generation process, their scheme requires the data owner to communicate with the data consumer to gain the essential information, which destroys the practicality of the model. In addition, the data owner must produce an ABE ciphertext during the re-encryption key generation. While the ABE encryption operation is computationally intensive, this scheme can not apply to resource-limited devices in the IIoT. In Table 1, we provide the comparison between related works.

Although the above studies have been successful in reducing users' computational cost, realizing fine-grained access control, and achieving flexible data sharing, they cannot be effectively implemented simultaneously. Thus, addressing the aforementioned issues in the IIoT environment remains a challenge.

1.2 Our Contribution

To address the aforementioned issues, we design an efficient data sharing scheme for the IIoT to achieve fine-grained access control and flexible data sharing while lowering the computational cost, which applies to data users with limited resources in the IIoT. The following are contributions of our study:

- We propose a non-interactive IBE-ABE PRE scheme that reduces the encryption computation cost of a data owner and achieves fine-grained access control of ciphertext data.
- Our scheme realizes flexible data sharing. Data owner can transform IBE encrypted ciphertext into ABE encrypted ciphertext. Thus, a group of data consumers with attributes matching the access policy can access the data. Simultaneously, it allows a data owner to make some changes to the access policy dynamically, realizing flexible data sharing.
- A formal security proof indicates the proposed scheme's selective chosen-plaintext attack (CPA) security. In addition, theoretical and experimental analyses illustrate that our construction is more effective than the previous schemes.

1.3 Organization

The remainder of this paper is organized as follows. Section 2 introduces the preliminaries of the proposed scheme. The system and security models are described in Section 3. In Section 4, we provide a detailed description of the proposed scheme. Section 5 presents the correctness analysis and security proof. In Section 6, we compare the performance of our scheme with those of other schemes. Finally, the conclusions of this study are presented in Section 7.

2 Preliminaries

2.1 Bilinear Maps

Definition 1. Let $\mathbb{G}$ and $\mathbb{G}_T$ represent two multiplicative cyclic groups with prime order p. The bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ has three attributes:

- Bilinearity: $\forall u_1, u_2 \in \mathbb{G}$, v and $w \in \mathbb{Z}_p$, we have $e(u_1^v, u_2^w) = e(u_1, u_2)^{vw}$.
- Non-degeneracy: $\exists u_1, u_2 \in \mathbb{G}$, $e(u_1, u_2) \neq 1$.
- Computability: $\forall u_1, u_2 \in \mathbb{G}$, $e(u_1, u_2)$ can be effectively computed.

Assume $\mathscr{G}$ is an effective bilinear group generation algorithm. The bilinear generator $\mathscr{G}$ returns a tuple $(\mathbb{G}, \mathbb{G}_T, p, e)$ when given a security parameter λ as input.

2.2 Access Structures

Definition 2. Let $\{U_1, U_2, \ldots, U_n\}$ represents a group parties, and access structure $\mathbb{A}$ is a non-empty subsets of $\{U_1, U_2, \ldots, U_n\}$, i.e., $2^{\{U_1, U_2, \ldots, U_n\}} \setminus \{\emptyset\}$. A collection $\mathbb{A}$ is monotonic if it satisfies the property: $\forall B$, C: if $B \in \mathbb{A}$, $B \subseteq C$, then $C \in \mathbb{A}$. A set in $\mathbb{A}$ is referred to as an authorized set, whereas a set not in $\mathbb{A}$ is referred to as an unauthorized set.

2.3 Linear Secret Sharing Schemes (LSSS)

Definition 3. The secret sharing scheme $\prod$ for a group of parties is defined as linear on $\mathbb{Z}_p$ if

1) The shares for each parity can comprise a vector on $\mathbb{Z}_p$.

2) There exists a share-generating matrix $M_{l \times n}$ for $\prod$. $\forall i \in [1, l]$, ρ is a function that maps each row of M to a specific parity and the ith row of M is associated with the $\rho(i)$. Suppose the vector $\vec{v} = (s, v_2, \ldots, v_n)^T \in_R \mathbb{Z}_p^n$, and s is the secret to be shared. According to $\prod$, the vector of l shares of the secret s is $M\vec{v}$. The share $(M\vec{v})_i$ relates to party $\rho(i)$.

Suppose $\prod$ is an LSSS associated with the access structure $\mathbb{A}$, let $S \in \mathbb{A}$ be any authorized set, and let $I \subset \{1, \ldots, l\}$ be defined as $I = \{i : \rho(i) \in S\}$. $\prod$ satisfies the property that for valid shares $\{\lambda_i\}$ of s, constants $\{\omega_i\}_{i \in I}$ can be computed in polynomial time such that $\sum_{i \in I} \omega_i \lambda_i = s$.

TABLE 2: Notations

Notation	Description
PK_{IBE}	IBE public key
PK_{ABE}	ABE public key
MSK_{IBE}	IBE master secret key
MSK_{ABE}	ABE master secret key
SK_{ID}	IBE private key
SK_S	ABE private key
TK	transformation key
RtK	retrieval key
CT_{ID}	original ciphertext
CT_A	re-encryption ciphertext
CT_S	transformed ciphertext
IK	intermediate re-encryption key
RK	re-encryption key

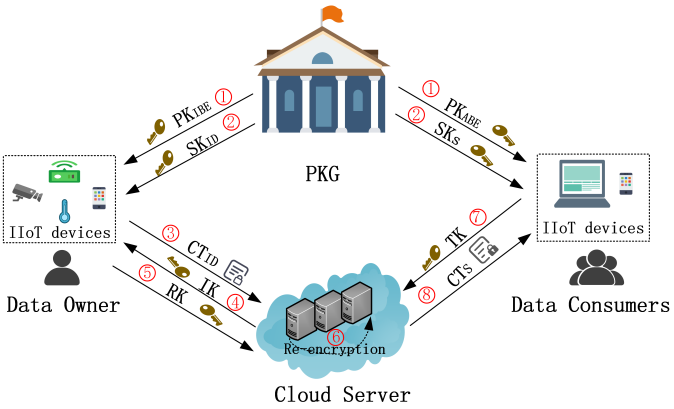


Fig. 2: System model

3 System Architecture

In this part, the system and security model are described in detail.

3.1 System Model

In the proposed system, there exist four entities: private key generator (PKG), cloud server, data owner, and data consumers. Table 5 includes the notations for use in our scheme. The organization of these entities is depicted in Fig. 2.

- PKG is in charge of setting up the system and generating private keys for data users.
- Data owner employs a collection of IIoT devices to gather data and perform some cryptographic computations. The data owner first encrypts the data and then uploads the resulting ciphertexts to the cloud server. He can also generate a re-encryption key and then delegate the cloud server to transform the ciphertext to previously unspecified data consumers. In the re-encryption key generation, complex computations are completed with the assistance of the cloud server.
- Cloud server is in charge of storing ciphertexts from the data owner, generating intermediate re-encryption keys for the data owner, responding to transformation requests from the data owner, and providing outsourced decryption services for data consumers.
- Data consumers use IIoT devices to perform decryption operations. Data consumers first obtain ciphertext from the cloud server and then use the retrieval key to recover the plaintext.

In our system, PKG initializes the system to generate public keys and master secret keys. It sends public keys PK_{IBE} and PK_{ABE} to users (step ①) and keeps master secret keys MSK_{IBE} and MSK_{ABE} confidential. When users want to join the system, PKG generates and issues private keys SK_{ID} and SK_S to users (step ②). To achieve data privacy protection, the data owner uses the IBE encryption mechanism to encrypt data and uploads the resulting ciphertext CT_{ID} to the cloud server (step ③). When the data owner desires to implement fine-grained data sharing with other data consumers, he can specify an access policy and delegate the cloud server to generate an intermediate re-encryption key IK related to the access policy (step ④). Then, the data owner combines the IK to generate a re-encryption key RK and sends the RK to the cloud server (step ⑤). After receiving the RK, the cloud server transforms the IBE encrypted ciphertext to the ABE encrypted ciphertext related to the access policy (step ⑥), allowing specified data consumers to access original data. When a data consumer with attributes that match the access policy of the ciphertext desires to access the original data, he can deliver a transformation key TK to the cloud server (step ⑦). The ciphertext can be partly decrypted by the cloud server using the TK, and then sends the transformed ciphertext CT_S to the data consumer (step ⑧), who uses the retrieval key RtK to recover the original data.

Based on the proposed system models, the system includes the following algorithms:

- $\mathbf{Setup_{IBE}}(\lambda) \rightarrow (PK_{IBE}, MSK_{IBE})$: It is performed by PKG. Given a security parameter λ, it returns an IBE public key PK_{IBE} and an IBE master secret key MSK_{IBE}.
- $\mathbf{Setup_{ABE}}(\lambda, U) \rightarrow (PK_{ABE}, MSK_{ABE})$: It is performed by PKG. Given a security parameter λ and a universe description U used to define the maximum number of system attributes, it returns an ABE public key PK_{ABE} and an ABE master secret key MSK_{ABE}.
- $\mathbf{KeyGen_{IBE}}(ID, MSK_{IBE}, PK_{IBE}) \rightarrow SK_{ID}$: It is performed by PKG. Taking an identity ID, an IBE master secret key MSK_{IBE}, and an IBE public key PK_{IBE} as input, it returns a private key SK_{ID} related to ID.
- $\mathbf{KeyGen_{ABE}}(S, MSK_{ABE}, PK_{ABE}) \rightarrow SK_S$: It is performed by PKG. Taking an attribute set S, an ABE master secret key MSK_{ABE} and an ABE public key PK_{ABE} as input, it returns a private key SK_S associated with S.
- $\mathbf{KeyGen_{out}}(SK_S) \rightarrow (TK, RtK)$: It is performed by data consumers. Taking an ABE private key SK_S as input, it returns a transformation key TK and a retrieval key RtK.
- $\mathbf{Encrypt_{IBE}}(ID, PK_{IBE}, \mathcal{M}) \rightarrow CT_{ID}$: It is performed by the data owner. Taking an identity ID, an IBE public key PK_{IBE}, and plaintext $\mathcal{M} \in \mathbb{G}_T$ as

input, it returns an IBE ciphertext CT_{ID}.

- $\mathbf{RKGen_{out}}(PK_{ABE}, (M, \rho)) \rightarrow IK$: It is performed by the cloud server. Taking an ABE public key PK_{ABE} and an LSSS access structure (M, ρ) related to a set of data consumers as input, it returns an intermediate re-encryption key IK.
- $\mathbf{RKGen_{user}}(PK_{ABE}, PK_{IBE}, SK_{ID}, IK) \rightarrow RK$: It is performed by the data owner. Taking the ABE and IBE public key PK_{ABE} and PK_{IBE}, an IBE private key SK_{ID}, and an intermediate re-encryption key IK as input, it returns a re-encryption key RK.
- $\mathbf{ReEnc}(RK, CT_{ID}) \rightarrow CT_A$: It is performed by the cloud server. Taking a re-encryption key RK and an IBE ciphertext CT_{ID} as input, it returns a re-encrypted ciphertext CT_A associated with the access structure (M, ρ).
- $\mathbf{Decrypt_{out}}(TK, CT_A) \rightarrow CT_S/\perp$: It is performed by the cloud server. Taking a transformation key TK and a ciphertext CT_A as input, and it returns a transformed ciphertext CT_S or the error message $\perp$.
- $\mathbf{Decrypt_{user}}(PK_{ABE}, RtK, CT_S) \rightarrow \mathcal{M}$: It is performed by data consumers. Taking an ABE public key PK_{ABE}, a retrieval key RtK, and a transformed ciphertext CT_S as input, it returns the plaintext $\mathcal{M}$.

Correctness. The correctness states that, for $\forall\ CT_A \leftarrow ReEnc(RK, CT_{ID})$, where $RK \leftarrow RKGen_{user}(PK_{ABE}, PK_{IBE}, SK_{ID}, IK)$, $IK \leftarrow RKGen_{out}(PK_{ABE}, (M, \rho))$, $CT_{ID} \leftarrow Encrypt_{IBE}(ID, PK_{IBE}, \mathcal{M})$, and any private key $SK_S \leftarrow KeyGen_{ABE}(S, MSK_{ABE}, PK_{ABE})$ and $(TK, RtK) \leftarrow KeyGen_{out}(SK_S)$, if S matches (M, ρ), the outsourced decryption algorithm $Decrypt_{out}(TK, CT_A)$ always outputs the correct transformed ciphertext CT_S, and the decryption algorithm $Decrypt_{user}(PK_{ABE}, RtK, CT_S)$ always recovers the plaintext $\mathcal{M}$.

For a re-encrypted ciphertext, the correctness defines that the ciphertext can be correctly re-encryption from the original IBE ciphertext, if the re-encryption key used in the re-encryption is created by a user who can decrypt the original IBE ciphertext. Also, the correctness defines that the re-encrypted ciphertext can be decrypted by data consumers with attributes that match the access structure.

3.2 Secutity Model

In our scheme, assuming that the cloud server is "honest-but-curious," that implies it will obey the protocol while attempting to obtain as many sensitive details as possible. We consider realistic attacks by unauthorized users, who may collude with the cloud server to try to acquire original data. Therefore, we simulate an adversary who can query both the ABE and IBE public keys and private keys. Besides, the adversary can request the selected attribute sets and identities for the intermediate re-encryption keys, re-encryption keys and re-encrypton ciphertexts. We consider two selective CPA security games, depending on the adversary's attack target.

Game GAME-Or. Aiming at the original ciphertext, the proposed scheme is selective CPA-secure if there has an adversary $\mathcal{A}$ with a negligible probability of breaking the following game.

Init. $\mathcal{A}$ selects a challenge identity ID^* and a challenge access structure $\mathbb{A}^*$, and outputs them to the challenger $\mathcal{C}$.

Setup. $\mathcal{C}$ executes $Setup_{IBE}$ and $Setup_{ABE}$ and returns PK_{IBE} and PK_{ABE} to $\mathcal{A}$.

Phase I. $\mathcal{A}$ queries:

- $Extract_{SK_{IBE}}(ID)$: $\mathcal{A}$ performs a query for ID if $ID \neq ID^*$, $\mathcal{C}$ executes $SK_{ID} = KeyGen_{IBE}(MSK_{IBE}, PK_{IBE}, ID)$ and returns SK_{ID} to $\mathcal{A}$.
- $Extract_{SK_S}(S)$: $\mathcal{A}$ performs a query for S if $S \notin \mathbb{A}^*$, $\mathcal{C}$ executes $SK_S = KeyGen_{ABE}(MSK_{ABE}, PK_{ABE}, S)$ and returns SK_S to $\mathcal{A}$.
- $Extract_{TK}(S)$: $\mathcal{A}$ queries a transformation key for $S \notin \mathbb{A}^*$, $\mathcal{C}$ executes $SK_S = KeyGen_{ABE}(MSK_{ABE}, PK_{ABE}, S)$ and runs $KeyGen_{out}(SK_S)$ to obtain TK, sends TK to $\mathcal{A}$.
- $Extract_{IK}(\mathbb{A})$: $\mathcal{A}$ queries an intermediate re-encryption key, $\mathcal{C}$ runs $IK = RKGen_{out}(PK_{ABE}, \mathbb{A})$ and returns IK to $\mathcal{A}$.
- $Extract_{RK}(ID, \mathbb{A})$: $\mathcal{A}$ queries a re-encryption key, $\mathcal{C}$ runs $IK = RKGen_{out}(PK_{ABE}, \mathbb{A})$, $SK_{ID} = KeyGen_{IBE}(ID, MSK_{IBE}, PK_{IBE})$ and $RK = RKGen_{user}(PK_{ABE}, PK_{IBE}, SK_{ID}, IK)$, returns RK to $\mathcal{A}$.
- $Extract_{Re}(CT_{ID}, ID, \mathbb{A})$: $\mathcal{A}$ performs a re-encryption query on $(CT_{ID}, ID, \mathbb{A})$, $\mathcal{C}$ runs $IK = RKGen_{out}(PK_{ABE}, \mathbb{A})$, $SK_{ID} = KeyGen_{IBE}(ID, MSK_{IBE}, PK_{IBE})$, $RK = RKGen_{user}(PK_{ABE}, PK_{IBE}, SK_{ID}, IK)$ and $CT_A = ReEnc(RK, CT_{ID})$, returns CT_A to $\mathcal{A}$.

In Phase I, $\mathcal{A}$ is not permitted to perform the following queries:

- $Extract_{SK_{IBE}}(ID)$ if $ID = ID^*$;
- $Extract_{SK_S}(S)$, where $S \in \mathbb{A}^*$;
- $Extract_{TK}(S)$, where $S \in \mathbb{A}^*$;
- $Extract_{RK}(ID, \mathbb{A})$ if $ID = ID^*$ and $\mathbb{A}$ has queried $Extract_{SK}(S)$ where $S \in \mathbb{A}$ and $\mathbb{A}$ is an arbitrary access structure.

Challenge. $\mathcal{A}$ chooses a message tuple $(\mathcal{M}_0, \mathcal{M}_1)$ satisfies $|\mathcal{M}_0| = |\mathcal{M}_1|$ and outputs it to $\mathcal{C}$. $\mathcal{C}$ randomly selects $\varphi \in \{0, 1\}$ and sets the challenge ciphertext $CT^*_{ID} = Encrypt_{IBE}(ID^*, PK_{IBE}, \mathcal{M}_\varphi)$. It returns CT^*_{ID} to $\mathcal{A}$.

Phase II. $\mathcal{A}$ performs requests as in Phase I apart from:

- $Extract_{SK_{IBE}}(ID)$ if $ID = ID^*$;
- $Extract_{SK_S}(S)$, where $S \in \mathbb{A}^*$;
- $Extract_{TK}(S)$, where $S \in \mathbb{A}^*$;
- $Extract_{RK}(ID, \mathbb{A})$, $Extract_{SK_S}(S)$ and $Extract_{TK}(S)$, where $ID = ID^*$, $S \in \mathbb{A}$.
- $Extract_{Re}(CT_{ID}, ID, \mathbb{A})$, $Extract_{SK_S}(S)$ and $Extract_{TK}(S)$, where $ID = ID^*$ and $S \in \mathbb{A}$.

Guess. $\mathcal{A}$ submits a guess $\varphi' \in \{0, 1\}$. $\mathcal{A}$ wins if $\varphi' = \varphi$.

In the above GAME-Or game, we define the advantage of $\mathcal{A}$ to be $Adv^{FGDS-Or}_{\mathcal{A}} = |Pr[\varphi' = \varphi] - 1/2|$.

Game GAME-Re. Aiming at the re-encrypted ciphertext, the proposed scheme is selective CPA-secure if there has an

adversary $\mathcal{A}$ with a negligible probability of breaking the following game. Init, Setup and Guess are the same as in GAME-Or. Phase I, Phase II and Challenge are as follows.

Phase I. $\mathcal{A}$ queries:

- $Extract_{SK_{IBE}}(ID)$: $\mathcal{A}$ performs a query for ID if $ID \neq ID^*$, $\mathcal{C}$ executes $SK_{ID} = KeyGen_{_{IBE}}(MSK_{IBE}, PK_{IBE}, ID)$ and returns SK_{ID} to $\mathcal{A}$.
- $Extract_{SK_S}(S)$: $\mathcal{A}$ performs a query for S if $S \notin \mathbb{A}^*$, $\mathcal{C}$ executes $SK_S = KeyGen_{_{ABE}}(MSK_{ABE}, PK_{ABE}, S)$ and returns SK_S to $\mathcal{A}$.
- $Extract_{TK}(S)$: $\mathcal{A}$ queries a transformation key for $S \notin \mathbb{A}^*$, $\mathcal{C}$ executes $SK_S = KeyGen_{_{ABE}}(MSK_{ABE}, PK_{ABE}, S)$ and runs $KeyGen_{out}(SK_S)$ to obtain TK, sends TK to $\mathcal{A}$.
- $Extract_{IK}(\mathbb{A})$: $\mathcal{A}$ queries an intermediate re-encryption key, $\mathcal{C}$ runs $IK = RKGen_{out}(PK_{ABE}, \mathbb{A})$ and returns IK to $\mathcal{A}$.
- $Extract_{RK}(ID, \mathbb{A})$: $\mathcal{A}$ queries a re-encryption key, $\mathcal{C}$ runs $IK = RKGen_{out}(PK_{ABE}, \mathbb{A})$, $SK_{ID} = KeyGen_{_{IBE}}(ID, MSK_{IBE}, PK_{IBE})$ and $RK = RKGen_{user}(PK_{ABE}, PK_{IBE}, SK_{ID}, IK)$, returns RK to $\mathcal{A}$.
- $Extract_{Re}(CT_{ID}, ID, \mathbb{A})$: $\mathcal{A}$ performs a re-encryption query on $(CT_{ID}, ID, \mathbb{A})$, $\mathcal{C}$ runs $IK = RKGen_{out}(PK_{ABE}, \mathbb{A})$, $SK_{ID} = KeyGen_{_{IBE}}(ID, MSK_{IBE}, PK_{IBE})$, $RK = RKGen_{user}(PK_{ABE}, PK_{IBE}, SK_{ID}, IK)$ and $CT_A = ReEnc(RK, CT_{ID})$, returns CT_A to $\mathcal{A}$.

In Phase I, $\mathcal{A}$ is not permitted to perform the following queries:

- $Extract_{SK_{IBE}}(ID)$ if $ID = ID^*$;
- $Extract_{SK_S}(S)$, where $S \in \mathbb{A}^*$;
- $Extract_{TK}(S)$, where $S \in \mathbb{A}^*$;

Challenge. $\mathcal{A}$ chooses a message tuple $(\mathcal{M}_0, \mathcal{M}_1)$ satisfies $|\mathcal{M}_0| = |\mathcal{M}_1|$ and outputs it to $\mathcal{C}$. $\mathcal{C}$ randomly selects $\varphi \in \{0, 1\}$ and calculates the challenge ciphertext $CT_A^* = ReEnc(RK, Encrypt_{_{IBE}}(ID, PK_{IBE}, \mathcal{M}_\varphi))$, where $RK = RKGen_{user}(ID, \mathbb{A}^*)$. It returns CT_A^* to $\mathcal{A}$.

Phase II. $\mathcal{A}$ performs requests as in Phase I apart from:

- $Extract_{SK_{IBE}}(ID)$ if $ID = ID^*$;
- $Extract_{SK_S}(S)$, where $S \in \mathbb{A}^*$;
- $Extract_{TK}(S)$, where $S \in \mathbb{A}^*$;

Guess. $\mathcal{A}$ submits a guess $\varphi' \in \{0, 1\}$. $\mathcal{A}$ wins if $\varphi' = \varphi$.

In the above GAME-Re game, we define the advantage of $\mathcal{A}$ to be $Adv_{\mathcal{A}}^{FGDS-Re} = |Pr[\varphi' = \varphi] - 1/2|$.

Definition 4. The proposed scheme is selective CPA-secure if GAME-Or and GAME-Re are secure.

4 Proposed Scheme

In this section, we instance the proposed scheme, which uses proxy re-encryption technology to realize the transformation from the ciphertext of an IBE scheme in [39] to the ciphertext of an ABE scheme in [40], and realizes efficient data sharing and fine-grained access control. And in this section, we describe our specific scheme in detail.

4.1 Setup

PKG initializes the system by calling the $Setup_{_{IBE}}$ algorithm and the $Setup_{_{ABE}}$ algorithm.

$\mathbf{Setup_{IBE}}(\lambda)$: Given a security parameter λ, PKG executes the bilinear generator $\mathscr{G}$ to generate a tuple $(\mathbb{G}, \mathbb{G}_T, p, e)$. Then, PKG randomly chooses $g_0 \in \mathbb{G}$ as a generator and $g_2, g_3, h \in \mathbb{G}$. It randomly selects $\alpha_1 \in \mathbb{Z}_p$ and defines $g_1 = g_0^{\alpha_1}$. The IBE public and master secret key are respectively:

$$PK_{IBE} = (g_0, g_1, g_2, g_3, h) \quad MSK_{IBE} = \alpha_1.$$

$\mathbf{Setup_{ABE}}(\lambda, U)$: Given a security parameter λ and a universe set U. First, PKG performs the bilinear generator $\mathscr{G}$ to generate a tuple $(\mathbb{G}, \mathbb{G}_T, p, e)$. Then, PKG selects $g \in \mathbb{G}$ as a generator. It randomly selects a group value $h_i \in \mathbb{G}$ for each attribute $i \in U$. PKG also selects random elements $\alpha_2, a \in \mathbb{Z}_p$. Besides, PKG selects a cryptographic hash function $\mathcal{F} : \mathbb{G}_T \to \mathbb{G}$. The ABE public and master secret key are respectively:

$$PK_{ABE} = \left\{g, g^a, e(g, g)^{\alpha_2}, \{h_i\}_{i \in U}, \mathcal{F}\right\} MSK_{ABE} = g^{\alpha_2}.$$

4.2 Key Generation

This phase consists of two algorithms: $KeyGen_{_{IBE}}$ and $KeyGen_{_{ABE}}$. The $KeyGen_{_{IBE}}$ algorithm and the $KeyGen_{_{ABE}}$ algorithm generate private keys for data users (data owner or data consumers).

$\mathbf{KeyGen_{IBE}}(ID, MSK_{IBE}, PK_{IBE})$: Taking an identity ID, an IBE master secret key MSK_{IBE}, and an IBE public key PK_{IBE} as input, PKG randomly selects $u \in \mathbb{Z}_p$, then calculates:

$$SK_{ID}^1 = g_2^{\alpha_1}(g_1^{ID}h)^u, \quad SK_{ID}^2 = g_0^u.$$

PKG sets the IBE private key as $SK_{ID} = (SK_{ID}^1, SK_{ID}^2)$.

$\mathbf{KeyGen_{ABE}}(S, MSK_{ABE}, PK_{ABE})$: Taking an attribute set S, an ABE master secret key MSK_{ABE}, and an ABE public key PK_{ABE} as input. PKG chooses $t \in \mathbb{Z}_p$ at random, then computes:

$$K = g^{\alpha_2}g^{at}, \quad L = g^t, \quad \forall x \in S : K_x = h_x^t.$$

PKG outputs the ABE private key $SK_S = (K, L, \{K_x\}_{x \in S})$.

4.3 Data Encryption

Before uploading data to the cloud server, the data owner calls the $Encrypt_{_{IBE}}$ algorithm to encrypt the data.

$\mathbf{Encrypt_{IBE}}(ID, PK_{IBE}, \mathcal{M})$: Taking an identity ID, an IBE public key PK_{IBE} and plaintext $\mathcal{M} \in \mathbb{G}_T$ as input, the data owner randomly chooses $w \in \mathbb{Z}_p$ and executes:

$$C_1 = g_0^w, C_2 = (g_1^{ID}h)^w, C_3 = \mathcal{M} \cdot e(g_1, g_2)^w, C_4 = g_3^w.$$

It sets an IBE ciphertext as $CT_{ID} = (C_1, C_2, C_3, C_4)$.

The ciphertext CT_{ID} is the same as the BB-IBE ciphertext in [39] if the component C_4 does not exist. If users want to share date with other users, they need to compute C_4 when encrypting the data. The component C_4 is not required during IBE decryption, but is necessary for re-encryption.

4.4 Intermediate Re-encryption Key Generation

In this phase, the cloud server generates an intermediate re-encryption key to reduce the computing cost of the data owner.

$\mathbf{RKGen_{out}}(PK_{ABE}, (M, \rho))$: Taking an ABE public key PK_{ABE} and an LSSS access structure (M, ρ) connected with a collection of data consumers as input, where the matrix M has l rows and n columns, and ρ is a function that maps each row of M to attributes. First, the cloud server selects $\tau' \in \mathbb{Z}_p$ at random, and calculates $d_0 = g^{\tau'}$. Then, the cloud server randomly chooses $\lambda_i', r_i' \in \mathbb{Z}_p$ for $i = 1$ to l and executes:

$$d_{i,1} = g^{a\lambda_i'} h_{\rho(i)}^{-r_i'}, \quad d_{i,2} = g^{r_i'}.$$

It sets intermediate re-encryption key as $IK = ((M, \rho), \tau', d_0, \{\lambda_i', r_i', d_{i,1}, d_{i,2}\}_{i\in[1,l]})$.

4.5 Re-encryption Key Generation

In this step, the data owner combines IK to generate a re-encryption key.

$\mathbf{RKGen_{user}}(PK_{ABE}, PK_{IBE}, SK_{ID}, IK)$: Taking the ABE and IBE public key PK_{ABE} and PK_{IBE}, an IBE private key SK_{ID} of the data owner, and an intermediate re-encryption key IK as input. The data owner randomly chooses $t', \tau \in \mathbb{Z}_p$ and executes:

$$d_3 = SK_{ID}^1 \cdot g_3^{t'}, \quad d_4 = SK_{ID}^2.$$

and

$$d_5 = \mathcal{F}(e(g,g)^{\alpha_2 \tau}) \cdot g_0^{t'}, \quad d_6 = \tau - \tau'.$$

Then, it selects $v_2, \ldots, v_n \in \mathbb{Z}_p$ at random, creates the vector $\vec{v} = (\tau, v_2, \ldots, v_n)^T$, and calculates shares of τ as $(\lambda_1, \ldots, \lambda_l)^T = M\vec{v}$. For $i = 1$ to l, it randomly selects $r_i \in \mathbb{Z}_p$ and calculates:

$$d_{i,7} = \lambda_i - \lambda_i', \quad d_{i,8} = r_i - r_i'.$$

This will correct to the shares of τ and randomize r_i. Set the re-encryption key as $RK = (d_0, d_3, d_4, d_5, d_6, \{d_{i,1}, d_{i,2}, d_{i,7}, d_{i,8}\}_{i\in[1,l]})$ and give it to the cloud server.

4.6 Re-encryption

After receiving the RK, the cloud server uses the $ReEnc$ algorithm to re-encrypt the original ciphertext.

$\mathbf{ReEnc}(RK, CT_{ID})$: Taking a $RK = (d_0, d_3, d_4, d_5, d_6, \{d_{i,1}, d_{i_2}, d_{i_7}, d_{i_8}\}_{i\in[1,l]})$ and a ciphertext $CT_{ID} = (C_1, C_2, C_3, C_4)$ as input, the cloud server computes:

$$E = \frac{e(d_4, C_2)}{e(d_3, C_1)} = \frac{1}{e(g_3^{t'}, g_0^w) e(g_1, g_2)^w}.$$

Then it follows that:

$$C' = C_3 \cdot E, C_0' = d_0, C_1' = d_5, C_{i,2}' = d_{i,1}, C_{i,3}' = d_{i,2},$$
$$C_4' = C_4, C_5' = d_6, C_{i,6}' = d_{i,7}, C_{i,7}' = d_{i,8}.$$

Finally, the re-encrypted ciphertext is set as $CT_A = (C', C_0, C_1', C_4', C_5', \left\{C_{i,2}', C_{i,3}', C_{i,6}', C_{i,7}'\right\}_{i\in[1,l]})$.

4.7 Transformation Key Generation

This phase generates a transformation key for partial decryption of ABE ciphertext.

$\mathbf{KeyGen_{out}}(SK_S)$: Taking an ABE private key SK_S as input, the data consumer randomly chooses $z \in \mathbb{Z}_p^*$, then computes:

$$K' = K^{1/z} = g^{\alpha_2/z} g^{at/z}, \quad L' = L^{1/z} = g^{t/z},$$
$$\forall x \in S : K_x' = K_x^{1/z} = h_x^{t/z}.$$

The transformation key as TK is $(K', L', \{K_x'\}_{x\in S})$, the retrieval key as RtK is z. It should be noted that the TK is delivered to the cloud server to decryption outsourcing, and the RtK is held by the data consumer to recover the plaintext.

4.8 Data Decryption

For reducing the decryption computation overhead for data consumers, we outsource complex computing to the cloud server to lessen their computing burden. There are two algorithms in this phase: $Decrypt_{out}$ and $Decrypt_{user}$.

$\mathbf{Decrypt_{out}}(TK, CT_A)$: Taking a transformation key $TK = (K', L', \{K_x'\}_{x\in S})$ connected with an attribute set S and a ciphertext $CT_A = (C', C_0, C_1', C_4', C_5', \left\{C_{i,2}', C_{i,3}', C_{i,6}', C_{i,7}'\right\}_{i\in[1,l]})$ associated with the access structure (M, ρ) as input. If S matches the access structure, it defines $I = \{i : \rho(i) \in S\}$ and calculates constants $\{\omega_i\}_{i\in I}$ to make $\sum_{i\in I} \omega_i \lambda_i = \tau$. The cloud server calculates:

$$A = \frac{e(C_0' \cdot g^{C_5'}, K')}{\prod_{i\in I}(e(C_{i,2}' \cdot g^{aC_{i,6}'} \cdot h_{\rho(i)}^{-C_{i,7}'}, L') e(C_{i,3}' \cdot g^{C_{i,7}'}, K_{\rho(i)}'))^{\omega_i}}$$
$$= e(g,g)^{\alpha_2 \tau / z}.$$

The transformed ciphertext is set as $CT_S = (C', C_1', A, C_4')$.

$\mathbf{Decrypt_{user}}(PK_{ABE}, RK, CT_S)$: Taking an ABE public key PK_{ABE}, a retrieval key RtK, and a transformed ciphertext CT_S as input, the data consumer executes:

$$g_0^{t'} = C_1'/\mathcal{F}(A^z), \quad \mathcal{M} = C' \cdot e(g_0^{t'}, C_4').$$

5 Security Analysis

In this section, we first indicate that our construction is correct and then present security proof of our scheme.

5.1 Correctness Analysis

We present the correctness of decrypting the re-encrypted ciphertext. In the process of re-encryption, if the identity ID of the RK equals the identity ID of the CT_{ID}, then the cloud server calculates:

$$E = \frac{e(d_4, C_2)}{e(d_3, C_1)} = \frac{e(g_0^u, (g_1^{ID} h)^w)}{e(g_2^{\alpha_1} (g_1^{ID} h)^u \cdot g_3^{t'}, g_0^w)}$$
$$= \frac{1}{e(g_3^{t'}, g_0^w) e(g_2^{\alpha_1}, g_0^w)} = \frac{1}{e(g_3^{t'}, g_0^w) e(g_1, g_2)^w}.$$

Then, the cloud server can compute:

$$C' = C_3 \cdot E = \frac{\mathcal{M} \cdot e(g_1, g_2)^w}{e(g_3^{t'}, g_0^w) e(g_1, g_2)^w}$$
$$= \mathcal{M}/e(g_3^{t'}, g_0^w).$$

In the decryption phrase, we first compute the value of intermediate decryption ciphertext. If the attribute set S of the TK matches the access structure $\mathbb{A}$ of the CT_A, then the cloud server calculates:

$$\begin{aligned}A &= \frac{e(C_0'\cdot g^{C_5'}, K')}{\prod_{i\in I}(e(C_{i,2}'\cdot g^{aC_{i,6}'}\cdot h_{\rho(i)}^{-C_{i,7}'}, L')e(C_{i,3}'\cdot g^{C_{i,7}'}, K_{\rho(i)}'))^{\omega_i}}\\ &= \frac{e(g^{\tau'}g^{\tau-\tau'}, g^{\alpha_2/z}g^{at/z})}{\prod_{i\in I}(e(g^{a\lambda_i'}h_{\rho(i)}^{-r_i}g^{a(\lambda_i-\lambda_i')}, g^{t/z})e(g^{r_i'}g^{r_i-r_i'}, h_{\rho(i)}^{t/z}))^{\omega_i}}\\ &= \frac{e(g^{\tau}, g^{\alpha_2/z}g^{at/z})}{\prod_{i\in I}(e(g^{a\lambda_i}h_{\rho(i)}^{-r_i}, g^{t/z})e(g^{r_i}, h_{\rho(i)}^{t/z}))^{\omega_i}}\\ &= \frac{e(g,g)^{\alpha_2\tau/z}e(g,g)^{at\tau/z}}{e(g,g)^{at\tau/z}}\\ &= e(g,g)^{\alpha_2\tau/z}.\end{aligned}$$

Then, the data consumer can compute:

$$C_1'/\mathcal{F}(A^z) = \frac{\mathcal{F}(e(g,g)^{\alpha_2\tau})\cdot g_0^{t'}}{\mathcal{F}((e(g,g)^{\alpha_2\tau/z})^z)} = g_0^{t'}.$$

It follows that

$$\mathcal{M} = C'\cdot e(g_0^{t'}, C_4') = \mathcal{M}/e(g_3^{t'}, g_0^w)\cdot e(g_0^{t'}, g_3^w).$$

5.2 Security Proof

In this section, the selective CPA-secure of the proposed scheme will be indicated.

Theorem 1. Aiming at the original ciphertext, the proposed scheme is selective CPA-secure if the assumption of decisional q-parallel BDHE [40] is held.

Proof. Assuming an adversary $\mathcal{A}$, whose advantage of breaking the GAME-Or security is non-negligible. We utilize $\mathcal{A}$ to create an algorithm $\mathcal{B}$ to solve the problem of decisional q-parallel BDHE. □

Initially, the lists of IBE private key, ABE private key, transformation key, intermediate re-encryption key, and re-encryption key are nothing.

- $List_{ID}$: stores the tuple (ID, SK_{ID}).
- $List_S$: stores the tuple (S, SK_S).
- $List_{TK}$: stores the tuple (S, TK).
- $List_{IK}$: stores the tuple $(\mathbb{A}, IK)$.
- $List_{RK}$: stores the tuple $(ID, \mathbb{A}, RK, number)$ with $number \in \{0,1\}$. RK is a right re-encryption key if $number = 1$, and RK is chosen randomly if $number = 0$.

Init. $\mathcal{A}$ selects a challenge identity ID^* and a challenge access policy $\mathbb{A}^*$, and outputs them to $\mathcal{B}$.

Setup. $\mathcal{B}$ simulates:

- IBE-Setup. $\mathcal{B}$ selects random elements f_1, f_2, f_3, $\vartheta \in \mathbb{Z}_p$, $g \in \mathbb{G}$ and executes $g_0 = g$, $g_1 = g^{af_1}$, $g_2 = g^{a^q f_2}$, $g_3 = g^{\vartheta}$, $h = {g_1}^{-ID^*}g^{f_3}$. Then, the master secret key is set as $MSK_{IBE} = af_1$. $\mathcal{B}$ outputs $PK_{IBE} = (g_0, g_1, g_2, g_3, h)$ to $\mathcal{A}$.
- ABE-Setup. $\mathcal{B}$ chooses random element β, and sets $e(g,g)^{\alpha_2} = e(g,g)^{\beta}\cdot e(g^a, g^{a^q})$. This equation means that $\alpha_2 = a^{q+1} + \beta$. For each attribute $x \in U$, $\mathcal{B}$ randomly chooses $t_x \in \mathbb{Z}_p$. X is the collection of indexes i and $\rho^*(i) = x$, $\mathcal{B}$ computes:

$$h_x = g^{t_x}\prod_{i\in X} g^{aM_{i,1}^*/b_i}\cdot g^{a^2M_{i,2}^*/b_i}\ldots g^{a^{n^*}M_{i,n^*}^*/b_i}$$

$\mathcal{B}$ sets $h_x = g^{t_x}$ if X is an empty set. $\mathcal{B}$ also chooses a cryptographic hash function $\mathcal{F}: \mathbb{G}_T \to \mathbb{G}$. Finally, $\mathcal{B}$ outputs $PK_{ABE} = (g, g^a, e(g,g)^{\alpha_2}, \{h_x\}_{\rho^*(i)\in U}, \mathcal{F})$ to $\mathcal{A}$.

Phase I. $\mathcal{A}$ queries:

- $Extract_{SK_{IBE}}(ID)$: $\mathcal{A}$ performs a query for ID if $ID \neq ID^*$. If the $List_{ID}$ contains (ID, SK_{ID}), $\mathcal{B}$ sends SK_{ID} to $\mathcal{A}$. Otherwise, $\mathcal{B}$ chooses a random value $u \in \mathbb{Z}_p$ and calculates:

$$SK_{ID}^1 = g^{\frac{-a^q f_2 f_3}{(ID-ID^*)}}(g^{af_1(ID-ID^*)}g^{f_3})^u$$
$$SK_{ID}^2 = g^{\frac{-a^q f_2}{(ID-ID^*)}}g^u$$

  $\mathcal{B}$ returns $SK_{ID} = (SK_{ID}^1, SK_{ID}^2)$ to $\mathcal{A}$ and adds (ID, SK_{ID}) to $List_{ID}$.
- $Extract_{SK_S}(S)$: $\mathcal{A}$ performs a query for S if $S \notin \mathbb{A}^*$. If the $List_S$ contains (S, SK_S), $\mathcal{B}$ sends SK_S to $\mathcal{A}$. Otherwise, $\mathcal{B}$ first forms a vector $\vec{v} = (v_1, \ldots, v_{n^*})$ with $v_1 = -1$ and then $\vec{v}\cdot M_i^* = 0$ for all i where $\rho^*(i) \in S$. $\mathcal{B}$ randomly selects $r \in \mathbb{Z}_p$ and calculates:

$$L = g^r\prod_{i=1}^{n^*}(g^{a^{q+1-i}})^{v_i} = g^t.$$

  This equation indicates that $t = r + \sum_{i=1}^{n^*} v_i a^{q+1-i}$. Then, $\mathcal{B}$ computes:

$$\begin{aligned}K &= g^{\alpha_2}g^{at}\\ &= g^{a^{q+1}+\beta}\cdot g^{ar+\sum_{i=1}^{n^*} v_i a^{q+2-i}}\\ &= g^{\beta}g^{ar}\prod_{j=2}^{n^*}(g^{a^{q+2-i}})^{v_i}.\end{aligned}$$

  $\mathcal{B}$ sets $K_x = L^{t_x}$ for $x \in S$, if the equation $\rho^*(i) = x$ is false for all i.
  X is the collection of all i, and for $x \in S$, $\rho^*(i) = x$. Then, $\mathcal{B}$ sets:

$$\begin{aligned}K_x &= h_x^t\\ &= L^{t_x}\prod_{i=X}\prod_{j=1}^{n^*}(g^{\frac{a^j\cdot r}{b_i}}\prod_{\substack{k=1\\k\neq j}}^{n^*}(g^{a^{q+1-j-k/b_i}})^{v_k})^{M_{i,j}^*}.\end{aligned}$$

  $\mathcal{B}$ returns $SK_S = (K, L, K_x)$ to $\mathcal{A}$ and adds (S, SK_S) to $List_S$.
- $Extract_{TK}(S)$: $\mathcal{A}$ queries a transformation key for $S \notin \mathbb{A}^*$. If the $List_{TK}$ contains (S, TK), $\mathcal{B}$ sends TK to $\mathcal{A}$. Otherwise, $\mathcal{B}$ makes a query $Extract_{SK_S}(S)$ to get SK_S and then computes TK with SK_S as in the $KeyGen_{out}$ algorithm. Finally, $\mathcal{B}$ adds (S, TK) to $List_{TK}$.
- $Extract_{IK}(\mathbb{A})$: $\mathcal{A}$ queries an intermediate re-encryption key for $\mathbb{A}$. If the $List_{IK}$ contains $(\mathbb{A}, IK)$, $\mathcal{B}$ sends IK to $\mathcal{A}$. Otherwise, $\mathcal{B}$ computes IK with

$\mathbb{A}$ as in the $RKGen_{out}$ algorithm. Finally, $\mathcal{B}$ adds $(\mathbb{A}, IK)$ to $List_{IK}$.

- $Extract_{RK}(ID, \mathbb{A})$: $\mathcal{A}$ queries a re-encryption key. If the $List_{RK}$ contains $(ID, \mathbb{A}, RK, number)$, $\mathcal{B}$ sends RK to $\mathcal{A}$. Otherwise, $\mathcal{B}$ executes:
  - If $ID = ID^*$ and there exists an entry (S, SK_S), where $S \in \mathbb{A}$, in $List_S$, outputs $\perp$.
  - Else if $ID = ID^*$ and no such an entry (S, SK_S), where $S \in \mathbb{A}$, exists in $List_S$, $\mathcal{B}$ chooses RK at random, and $(ID, \mathbb{A}, RK, 0)$ is added to the $List_{RK}$. Otherwise,
  - $\mathcal{B}$ queries $Extract_{SK_{IBE}}(ID)$ and $Extract_{IK}(\mathbb{A})$ to get SK_{ID} and IK, and then computes RK using SK_{ID} and IK as in the $RKGen_{user}$ algorithm. Finally, $(ID, \mathbb{A}, RK, 1)$ is added to the $List_{RK}$ by $\mathcal{B}$.
- $Extract_{Re}(CT_{ID}, ID, \mathbb{A})$: If $ID = ID^*$ and there exists an entry (S, SK_S), where $S \in \mathbb{A}$, in $List_S$, outputs $\perp$. Otherwise, $\mathcal{B}$ first obtains the RK through query $Extract_{RK}(ID, \mathbb{A})$ and then re-encrypts CT_{ID} using the RK.

Challenge. $\mathcal{A}$ chooses a message tuple $(\mathcal{M}_0, \mathcal{M}_1)$ satisfies $|\mathcal{M}_0| = |\mathcal{M}_1|$ and outputs it to $\mathcal{B}$. $\mathcal{B}$ selects $\varphi \in \{0, 1\}$ at random and executes:

$$C_1^* = \mathcal{M}_\varphi \cdot T^{f_1 f_2},\ C_2^* = g^s,\ C_3^* = g^{s f_3},\ C_4^* = C_2^{*\vartheta}.$$

Phase II. $\mathcal{A}$ performs requests as in Phase I, with limitations described in the game GAME-Or.

Guess. $\mathcal{A}$ submits a guess φ'. If $\varphi' = \varphi$ then $\mathcal{B}$ returns 1, indicating that T equals $e(g,g)^{a^{q+1}s}$. Else, it returns 0, indicating that T is chosen randomly from $\mathbb{G}_T$.

Analysis. If $T = e(g,g)^{a^{q+1}s}$, $C_1^* = \mathcal{M}_\varphi \cdot T^{f_1 f_2} = \mathcal{M}_\varphi \cdot e(g^{a f_1}, g^{a^q f_2})^s = \mathcal{M}_\varphi \cdot e(g_1, g_2)^s$, which is a correct form with $Pr[\mathcal{B}(\vec{y}, T) = 1] = 1/2 + \delta$. Else T is chosen randomly from $\mathbb{G}_T$, $Pr[\mathcal{B}(\vec{y}, T)] = 1/2$. Therefore, the advantage that $\mathcal{B}$ solves the assumption of decisional q-parallel BDHE is non-negligible.

Theorem 2. Aiming at the re-encrypted ciphertext, the proposed scheme is selective CPA-secure if the assumption of decisional q-parallel BDHE is held.

Proof. Assuming an adversary $\mathcal{A}$, whose advantage of breaking the GAME-Re security is non-negligible. We utilize $\mathcal{A}$ to create an algorithm $\mathcal{B}$ to solve the problem of decisional q-parallel BDHE. □

Initially, the lists of IBE private key, ABE private key, transformation key, intermediate re-encryption key, and re-encryption key are nothing.

- $List_{ID}$: stores the tuple (ID, SK_{ID}).
- $List_S$: stores the tuple (S, SK_S).
- $List_{TK}$: stores the tuple (S, TK).
- $List_{IK}$: stores the tuple $(\mathbb{A}, IK)$.
- $List_{RK}$: stores the tuple $(ID, \mathbb{A}, RK)$.

Init. $\mathcal{A}$ selects a challenge identity ID^* and a challenge access policy $\mathbb{A}^*$, and outputs them to $\mathcal{B}$.

Setup. Same as that of Theorem 1 proof.

Phase I. $\mathcal{A}$ queries:

- $Extract_{SK_{IBE}}(ID)$: Same as in the Theorem 1 proof.
- $Extract_{SK_S}(S)$: Same as in the Theorem 1 proof.
- $Extract_{TK}(S)$: Same as in the Theorem 1 proof.
- $Extract_{IK}(\mathbb{A})$: Same as in the Theorem 1 proof.
- $Extract_{RK}(ID, \mathbb{A})$: $\mathcal{A}$ queries a re-encryption key. If the $List_{RK}$ contains $(ID, \mathbb{A}, RK)$, $\mathcal{B}$ sends RK to $\mathcal{A}$. Otherwise, $\mathcal{B}$ executes:
  - If $ID = ID^*$, outputs $\perp$.
  - Else $\mathcal{B}$ first queries $Extract_{SK_{IBE}}(ID)$ and $Extract_{IK}(\mathbb{A})$ to get SK_{ID} and IK, and then computes RK using SK_{ID} and IK as in the $RKGen_{user}$ algorithm. Finally, $(ID, \mathbb{A}, RK)$ is added to the $List_{RK}$ by $\mathcal{B}$.
- $Extract_{Re}(CT_{ID}, ID, \mathbb{A})$: If $ID = ID^*$, outputs $\perp$. Otherwise, $\mathcal{B}$ first obtains RK through query $Extract_{RK}(ID, \mathbb{A})$ and then re-encrypts CT_{ID} using the RK.

Challenge. $\mathcal{A}$ chooses a message tuple $(\mathcal{M}_0, \mathcal{M}_1)$ satisfies $|\mathcal{M}_0| = |\mathcal{M}_1|$ and outputs it to $\mathcal{B}$. $\mathcal{B}$ first chooses an identity where $ID \neq ID^*$, and generates a private key SK_{ID} and intermediate re-encryption $IK = Extract_{IK}(\mathbb{A}^*)$. $\mathcal{B}$ computes RK using SK_{ID} and IK as in the $RKGen_{user}$ algorithm. Then, $\mathcal{B}$ computes $CT_{ID} = Enc(\mathcal{M}_\varphi, ID)$ as in the Challenge phase in Theorem 1 proof. Finally, $\mathcal{B}$ calculates $CT_A^* = ReEnc(RK, CT_{ID})$, and outputs CT_A^* to $\mathcal{A}$.

Phase II. $\mathcal{A}$ performs requests as in Phase I.

Guess. $\mathcal{A}$ submits a guess φ'. If $\varphi' = \varphi$ then $\mathcal{B}$ returns 1, indicating that T equals $e(g,g)^{a^{q+1}s}$. Else, it returns 0, indicating that T is chosen randomly from $\mathbb{G}_T$.

Analysis. If $T = e(g,g)^{a^{q+1}s}$, CT_A^* is a correct form with $Pr[\mathcal{B}(\vec{y}, T) = 1] = 1/2 + \delta$. Else T is chosen randomly from $\mathbb{G}_T$, $Pr[\mathcal{B}(\vec{y}, T)] = 1/2$. Therefore, the advantage that $\mathcal{B}$ solves the assumption of decisional q-parallel BDHE is non-negligible.

Thus, the proposed scheme is selective CPA-secure.

6 Performance Evaluations

In this part, the performance of the proposed scheme will be evaluated.

6.1 Theoretical Analysis

In our theoretical analysis: t_e represents the exponentiation computation, t_p represents the computation in bilinear pairings, $|\mathbb{Z}_p|$ represents the size of an element in $\mathbb{Z}_p$, $|\mathbb{G}|$ represents the size of group $\mathbb{G}$, $|\mathbb{G}_T|$ represents the size of group $\mathbb{G}_T$. The number of attributes contained in each algorithm is represented by x. Besides, we make no distinction in exponentiation operations of $\mathbb{G}_T$ and $\mathbb{G}$.

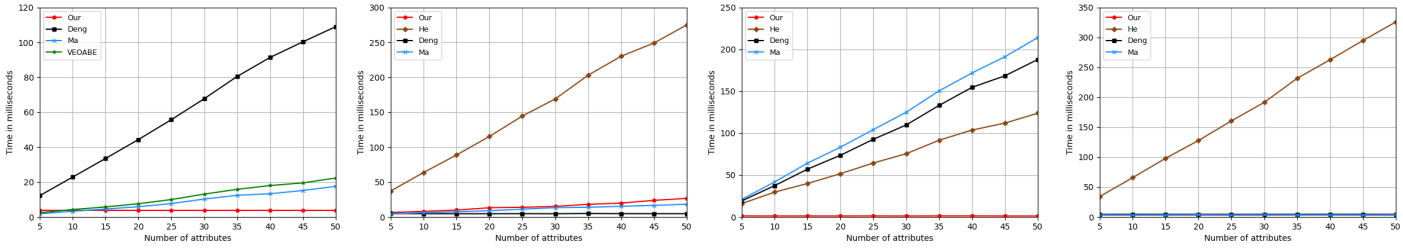
For transforming IBE encrypted ciphertext into ABE encrypted ciphertext, a trivial way is Decrypt-and-Re-Encrypt. And the data owner does not need to generate a re-encryption key and delegate the cloud server to perform the re-encryption operation. To this end, we compare the efficiency of this way with our scheme. Table 3 shows the efficiency comparison results between the two methods, the trivial Decrypt-and-Re-Encrypt way and the usage of PRE to transform the IBE encrypted ciphertext to the ABE

TABLE 3: Comparison with Trivial Decrypt and Re-Encrypt way

	Trivial Decrypt ane Re-encrypt	IBE→ABE
Computation	IBE.Dec+ABE.Enc: $2t_p + (2+3x)t_e$	RKGen (data owner side): $3t_e$ ReEnc(cloud server side): $2t_p$
Communication	(IBE.CT+ABE.CT)Size: $2\|\mathbb{G}_T\| + (3+2x)\|\mathbb{G}\|$	RK.Size (from data owner to cloud server): $(4+2x)\|\mathbb{G}\| + (1+2x)\|\mathbb{Z}_p\|$

TABLE 4: Comparison with related schemes

Schemes	Costs at Data Owner side			Costs at Cloud Server side		
	Public key storage	Private key storage	RK generation computation	Original ciphertext storage	Transformed ciphertext storage	Re-encryption computation
He *et al.* [26]	$4\|\mathbb{G}\|$	$2\|\mathbb{G}\|$	$(3x+8)t_e$	$2\|\mathbb{G}\| + \|\mathbb{G}_T\|$	$(2x+1)\|\mathbb{G}\| + 2\|\mathbb{G}_T\|$	$3t_p + (2x+2)t_e$
Deng *et al.* [35]	$6\|\mathbb{G}\| + \|\mathbb{G}_T\|$	$(2x+2)\|\mathbb{G}\|$	$6t_e$	$(3x+1)\|\mathbb{G}\| + \|\mathbb{G}_T\|$	$3\|\mathbb{G}\| + \|\mathbb{G}_T\|$	$(3x+1)t_p + xt_e$
Ma *et al.* [36]	$5\|\mathbb{G}\| + \|\mathbb{G}_T\|$	$(2x+2)\|\mathbb{G}\| + (x+1)\|\mathbb{Z}_p\|$	0	$(3x+1)\|\mathbb{G}\| + \|\mathbb{G}_T\| + 3x\|\mathbb{Z}_p\|$	$(3x+3)\|\mathbb{G}\| + \|\mathbb{G}_T\| + (2x+1)\|\mathbb{Z}_p\|$	$(3x+1)t_p + (3x+1)t_e$
Our	$4\|\mathbb{G}\|$	$2\|\mathbb{G}\|$	$3t_e$	$2\|\mathbb{G}\| + \|\mathbb{G}_T\|$	$(2x+3)\|\mathbb{G}\| + \|\mathbb{G}_T\| + (2x+1)\|\mathbb{Z}_p\|$	$2t_p$



(a) Encrypt.user Time (b) RkGen.user Time (c) ReEnc Time (d) Decrypt.user Time

Fig. 3: Experimental Results

encrypted ciphertext in our scheme. In the trivial Decrypt-and-Re-Encrypt way, the data owner first downloads IBE ciphertext data provided by the cloud server, then recovers the underlying data through the IBE private key, re-encrypt the data into ABE ciphertext form, and finally uploads ABE ciphertext to the cloud server. In terms of computing overhead, as seen in the table, the trivial way needs that the data owner spends linear cost in pairings. In our scheme, complex computing is done by the cloud server, while the data owner is only required to perform three exponentiation operations in the RK generation, which significantly reduces the computing overhead of the data owner. Although in terms of communication overhead, our scheme is slightly larger than the trivial Decrypt-and-Re-Encrypt way. But in general, our scheme has better performance than the trivial way.

Table 4 compares our scheme to others, including storage costs for the data owner and the cloud server, computational costs for generating RK and performing re-encryption. As depicted in Table 4, our scheme has the lowest cost on the data owner side, the storage cost of public and private keys is fixed, and the data owner is only required to execute three exponentiation operations in the RK generation. In He *et al.*'s scheme [26], the cost of RK generation is high, which increases the computational load on the data owner. Deng *et al.*'s scheme [35] and Ma *et al.*'s scheme [36] have low computational overhead to generate RK, but the data owner has a high storage overhead, and the size of the private key linear growth as the number of attributes grows. In terms of the cloud server cost, the storage cost of several schemes is almost the same. Our scheme has the smallest overhead in the RK generation, while the overhead of other schemes linear growth as the number of attributes grows, which increases the computing overhead of the cloud server.

6.2 Experimental Analysis

We performed a series of experimental analyses to compare the performance of our scheme with other related schemes. The schemes are performed in Ubuntu 20.04 LTS 64-bit by using Python 3.8 with the SS512 elliptic curve from the charm 0.50 framework on an Intel Core i5-7500 CPU @3.40 GHz with 16 GB RAM. The access policy's complexity influences the computational cost of these schemes. To consider the worst case, we set 'A_1 and A_2 and ... and A_n' as the access policy, which ensures that all attributes are included in this access policy. Where attributes are represented by any string.

We use the SS512 elliptic curve (a symmetric curve with a 512-bit base field and a security level of 80 bits) in the charm library to realize pairing operations. We implemented our scheme and three other comparison schemes, which are all based on the symmetric group. To reflect the effect of the number of attributes on the running time, we set the number of attributes from 0 to 50, and record the specific running time for each number of attributes. Considering the accuracy of the experimental analysis, the running time is calculated in our experiments by averaging the results of executing each operation 50 times. The time is given in milliseconds. The detailed experimental results are as follows.

1) *Encryption Time*: As depicted in Fig. 3a, Deng *et al.*'s scheme [35] has the largest encryption cost, because the encryption time of the ABE algorithm is linear growth as the number of attributes grows. Ma *et al.*'s scheme [36] and VEOABE [34] adopt outsourcing ABE encryption, and the complex encryption computation is completed by the cloud server, so the encryption overhead of the data owner is much less than Deng *et al.*'s scheme [35]. Our scheme adopts IBE encryption, so the data owner's encryption overhead is constant (about 3.85ms). When the number of attributes is small, the encryption overhead of Ma *et al.*'s scheme [36] and VEOABE [34] is slightly smaller than our scheme. However, when the number of attributes grows, their encryption overhead linear growth as the number of attributes grows and is significantly greater than our scheme. Therefore, our construction considerably minimizes the encryption computational overhead of the data owner through IBE encryption.

2) *RK Generation Time*: Fig. 3b shows that He *et al.*'s scheme [26] has the highest cost for generating the RK and the time overhead linear growth as the number of attributes grows. The time cost Ma *et al.*'s scheme [36] and our scheme is similar, and they both outsource part of the RK computation to the cloud server, lowering the data owner's computing cost. Deng *et al.*'s scheme [35] has the lowest cost to generate the RK because it transforms the ABE encrypted ciphertext to the IBE encrypted ciphertext, but it does not support efficient data sharing.

3) *Re-encryption Time*: As depicted in Fig. 3c, the re-encryption cost of our scheme is the smallest. The re-encryption cost of the other three schemes linear growth as the number of attributes grows, because they perform ABE decryption operation, while our scheme performs IBE decryption operation in the re-encryption algorithm.

4) *Decryption Time*: Fig. 3d shows that He *et al.*'s scheme [26] has the largest decryption cost, which linear growth as the number of attributes grows. In our scheme and Ma *et al.*'s scheme [36], the data owner's computing cost is lowed by outsourcing the decryption operation and the decryption overhead is almost the IBE decryption overhead of Deng *et al.*'s scheme [35].

The above experimental results indicate that our construction is effective and practical. Through outsourcing encryption and decryption technology and using the simple characteristics of IBE encryption operation, the computing overhead of the data users is significantly reduced and fine-grained data sharing is realized.

TABLE 5: Function comparison with related schemes

Schemes	CDT	NIT	EDS
He *et al.* [26]	√	×	√
Deng *et al.* [35]	√	√	×
Ma *et al.* [36]	×	√	√
Our	√	√	√

6.3 Function comparison

Table 5 compare our scheme with other schemes in terms of cross-domain transformation (CDT), non-interactive transformation (NIT) and efficient data sharing (EDS). CDT here refers to transforming ciphertext in one encryption format into ciphertext in another encryption format. He *et al.*'s scheme [26] supports cross-domain transformation by using PRE to transform the IBE encrypted ciphertext to the ABE encrypted ciphertext, but this scheme does not support non-interactive transformation. In the RK generation, the data owner needs to communicate with the data consumer to gain the essential information, destroying the model's practicability. Deng *et al.* [35] transform the ABE encrypted ciphertext into the IBE encrypted ciphertext, but their scheme can not achieve efficient data sharing. If the data owner needs to share data with multiple data consumers, he needs to perform multiple ciphertext transformations. Ma *et al.*'s scheme [36] can not realize cross-domain transformation, and it can only be transformed into one encryption system. Compared with these schemes, only our scheme supports cross-domain transformation, non-interactive transformation, and efficient data sharing.

7 Conclusion

In this paper, we designed a fine-grained data sharing scheme for resource-constrained devices in the IIoT, which not only accomplishes flexible data sharing and fine-grained access control, but also lowers data users' computational cost. We used PRE technology to transform the IBE encrypted ciphertext to the ABE encrypted ciphertext, realizing non-interactive transformation. In the re-encryption key generation, the cloud server generated an intermediate re-encryption key, which reduces the computing overhead of the data owner. Moreover, we outsourced most decryption operations to the cloud server, reducing the decryption computing overhead of data consumers. A formal security proof verified the proposed data sharing scheme's security. Theoretical and experimental analyses showed that our scheme's effectiveness and flexibility, and adaptability to devices with limited resources in the IIoT.

References

[1] M. R. Palattella, M. Dohler, A. Grieco, G. Rizzo, J. Torsner, T. Engel, and L. Ladid, "Internet of things in the 5g era: Enablers, architecture, and business models," *IEEE journal on selected areas in communications*, vol. 34, no. 3, pp. 510–527, 2016.

[2] E. Sisinni, A. Saifullah, S. Han, U. Jennehag, and M. Gidlund, "Industrial internet of things: Challenges, opportunities, and directions," *IEEE transactions on industrial informatics*, vol. 14, no. 11, pp. 4724–4734, 2018.

[3] J. Cui, F. Wang, Q. Zhang, Y. Xu, and H. Zhong, "An anonymous message authentication scheme for semi-trusted edge-enabled iiot," *IEEE Transactions on Industrial Electronics*, 2020.

[4] L. Wei, J. Cui, H. Zhong, I. Bolodurina, and L. Liu, "A lightweight and conditional privacy-preserving authenticated key agreement scheme with multi-ta model for fog-based vanets," *IEEE Transactions on Dependable and Secure Computing*, 2021.

[5] J.-S. Fu, Y. Liu, H.-C. Chao, B. K. Bhargava, and Z.-J. Zhang, "Secure data storage and searching for industrial iot by integrating fog computing and cloud computing," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 10, pp. 4519–4528, 2018.

[6] W. Wang, P. Xu, and L. T. Yang, "Secure data collection, storage and access in cloud-assisted iot," *IEEE cloud computing*, vol. 5, no. 4, pp. 77–88, 2018.

[7] Z. Li, Z. Yang, and S. Xie, "Computing resource trading for edge-cloud-assisted internet of things," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 6, pp. 3661–3669, 2019.

[8] S. Chaudhry, "An encryption-based secure framework for data transmission in iot," in *2018 7th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions)(ICRITO)*. IEEE, 2018, pp. 743–747.

[9] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Annual international conference on the theory and applications of cryptographic techniques*. Springer, 2005, pp. 457–473.

[10] J. Cui, B. Li, H. Zhong, G. Min, Y. Xu, and L. Liu, "A practical and efficient bidirectional access control scheme for cloud-edge data sharing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 2, pp. 476–488, 2021.

[11] K. Xue, N. Gai, J. Hong, D. Wei, P. Hong, and N. Yu, "Efficient and secure attribute-based access control with identical sub-policies frequently used in cloud storage," *IEEE Transactions on Dependable and Secure Computing*, 2020.

[12] Q. Zhang, H. Zhong, J. Cui, L. Ren, and W. Shi, "Ac4av: A flexible and dynamic access control framework for connected and autonomous vehicles," *IEEE Internet of Things Journal*, vol. 8, no. 3, pp. 1946–1958, 2020.

[13] S. Qi, Y. Lu, W. Wei, and X. Chen, "Efficient data access control with fine-grained data protection in cloud-assisted iiot," *IEEE Internet of Things Journal*, vol. 8, no. 4, pp. 2886–2899, 2020.

[14] S. Hohenberger and B. Waters, "Online/offline attribute-based encryption," in *International workshop on public key cryptography*. Springer, 2014, pp. 293–310.

[15] M. Green, S. Hohenberger, B. Waters *et al.*, "Outsourcing the decryption of abe ciphertexts." in *USENIX security symposium*, vol. 2011, no. 3, 2011.

[16] P. Zhang, Z. Chen, J. K. Liu, K. Liang, and H. Liu, "An efficient access control scheme with outsourcing capability and attribute update for fog computing," *Future Generation Computer Systems*, vol. 78, pp. 753–762, 2018.

[17] H. Zhong, Y. Zhou, Q. Zhang, Y. Xu, and J. Cui, "An efficient and outsourcing-supported attribute-based access control scheme for edge-enabled smart healthcare," *Future Generation Computer Systems*, vol. 115, pp. 486–496, 2021.

[18] M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography," in *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 1998, pp. 127–144.

[19] S. Kim and I. Lee, "Iot device security based on proxy re-encryption," *Journal of Ambient Intelligence and Humanized Computing*, vol. 9, no. 4, pp. 1267–1273, 2018.

[20] G. Pareek and B. Purushothama, "Proxy re-encryption for fine-grained access control: Its applicability, security under stronger notions and performance," *Journal of Information Security and Applications*, vol. 54, p. 102543, 2020.

[21] W. Zhang, H. Zhang, L. Fang, Z. Liu, and C. Ge, "A secure revocable fine-grained access control and data sharing scheme for scada in iiot systems," *IEEE Internet of Things Journal*, vol. 9, no. 3, pp. 1976–1984, 2021.

[22] K. Liang and W. Susilo, "Searchable attribute-based mechanism with efficient data sharing for secure cloud storage," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 9, pp. 1981–1992, 2015.

[23] C. Ge, W. Susilo, J. Baek, Z. Liu, J. Xia, and L. Fang, "A verifiable and fair attribute-based proxy re-encryption scheme for data sharing in clouds," *IEEE Transactions on Dependable and Secure Computing*, 2021.

[24] K. Liang, M. H. Au, J. K. Liu, W. Susilo, D. S. Wong, G. Yang, Y. Yu, and A. Yang, "A secure and efficient ciphertext-policy attribute-based proxy re-encryption for cloud data sharing," *Future Generation Computer Systems*, vol. 52, pp. 95–108, 2015.

[25] K. Yu, L. Tan, M. Aloqaily, H. Yang, and Y. Jararweh, "Blockchain-enhanced data sharing with traceable and direct revocation in iiot," *IEEE transactions on industrial informatics*, vol. 17, no. 11, pp. 7669–7678, 2021.

[26] K. He, Y. Mao, J. Ning, K. Liang, X. Huang, E. Panaousis, and G. Loukas, "A new encrypted data switching protocol: Bridging ibe and abe without loss of data confidentiality," *IEEE Access*, vol. 7, pp. 50 658–50 668, 2019.

[27] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM conference on Computer and communications security*, 2006, pp. 89–98.

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

[29] J. Feng, H. Xiong, J. Chen, Y. Xiang, and K.-H. Yeh, "Scalable and revocable attribute-based data sharing with short revocation list for iiot," *IEEE Internet of Things Journal*, 2022.

[30] J. Sun, Y. Yuan, M. Tang, X. Cheng, X. Nie, and M. U. Aftab, "Privacy-preserving bilateral fine-grained access control for cloud-enabled industrial iot healthcare," *IEEE Transactions on Industrial Informatics*, 2021.

[31] F. Guo, Y. Mu, and Z. Chen, "Identity-based online/offline encryption," in *International conference on financial cryptography and data security*. Springer, 2008, pp. 247–261.

[32] H. Wang, D. He, and J. Han, "Vod-adac: Anonymous distributed fine-grained access control protocol with verifiable outsourced decryption in public cloud," *IEEE transactions on services computing*, vol. 13, no. 3, pp. 572–583, 2017.

[33] R. Zhang, H. Ma, and Y. Lu, "Fine-grained access control system based on fully outsourced attribute-based encryption," *Journal of Systems and Software*, vol. 125, pp. 344–353, 2017.

[34] H. Ma, R. Zhang, Z. Wan, Y. Lu, and S. Lin, "Verifiable and exculpable outsourced attribute-based encryption for access control in cloud computing," *IEEE transactions on dependable and secure computing*, vol. 14, no. 6, pp. 679–692, 2015.

[35] H. Deng, Z. Qin, Q. Wu, Z. Guan, and Y. Zhou, "Flexible attribute-based proxy re-encryption for efficient data sharing," *Information Sciences*, vol. 511, pp. 94–113, 2020.

[36] H. Ma, R. Zhang, G. Yang, Z. Song, K. He, and Y. Xiao, "Efficient fine-grained data sharing mechanism for electronic medical record systems with mobile devices," *IEEE Transactions on Dependable and Secure Computing*, vol. 17, no. 5, pp. 1026–1038, 2018.

[37] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *Annual international cryptology conference*. Springer, 2001, pp. 213–229.

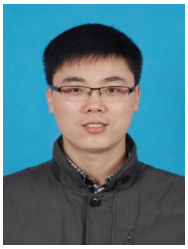
[38] J. Shao, R. Lu, and X. Lin, "Fine-grained data sharing in cloud computing for mobile devices," in *2015 IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 2015, pp. 2677–2685.

[39] D. Boneh and X. Boyen, "Secure identity based encryption without random oracles," in *Annual International Cryptology Conference*. Springer, 2004, pp. 443–459.

[40] B. Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization," in *International Workshop on Public Key Cryptography*. Springer, 2011, pp. 53–70.

Qiangyang Zhang was born in Anhui Province, China, in 1992. He received his B. Eng. degree and Ph.D. degree in computer science from Anhui University in 2021. He is currently a lecture of School of Computer Science and Technology at Anhui University. His research interest includes edge computing, computer systems, and security.

Yujie Fu is now a research student in the School of Computer Science and Technology, Anhui University. Her research focuses on the security of the Internet of Things.



Jie Cui (Senior Member, IEEE) was born in Henan Province, China, in 1980. He received his Ph.D. degree in University of Science and Technology of China in 2012. He is currently a professor and Ph.D. supervisor of the School of Computer Science and Technology at Anhui University. His current research interests include applied cryptography, IoT security, vehicular ad hoc network, cloud computing security and software-defined networking (SDN). He has over 150 scientific publications in reputable journals (e.g. IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Information Forensics and Security, IEEE Journal on Selected Areas in Communications, IEEE Transactions on Mobile Computing, IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Computers, IEEE Transactions on Vehicular Technology, IEEE Transactions on Intelligent Transportation Systems, IEEE Transactions on Network and Service Management, IEEE Transactions on Industrial Informatics, IEEE Transactions on Industrial Electronics, IEEE Transactions on Cloud Computing and IEEE Transactions on Multimedia), academic books and international conferences.

Debiao He received his Ph.D. degree in applied mathematics from School of Mathematics and Statistics, Wuhan University, Wuhan, China in 2009. He is currently a professor of the School of Cyber Science and Engineering, Wuhan University, Wuhan, China. His main research interests include cryptography and information security, in particular, cryptographic protocols. He has published over 100 research papers in refereed international journals and conferences, such as IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Information Security and Forensic, and Usenix Security Symposium. He is the recipient of the 2018 IEEE Systems Journal Best Paper Award and the 2019 IET Information Security Best Paper Award. His work has been cited more than 10000 times at Google Scholar. He is in the Editorial Board of several international journals, such as Journal of Information Security and Applications, Frontiers of Computer Science, and Human-centric Computing & Information Sciences.

Hong Zhong was born in Anhui Province, China, in 1965. She received her PhD degree in computer science from University of Science and Technology of China in 2005. She is currently a professor and Ph.D. supervisor of the School of Computer Science and Technology at Anhui University. Her research interests include applied cryptography, IoT security, vehicular ad hoc network, cloud computing security and software-defined networking (SDN). She has over 200 scientific publications in reputable journals (e.g. IEEE Journal on Selected Areas in Communications, IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Mobile Computing, IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Information Forensics and Security, IEEE Transactions on Intelligent Transportation Systems, IEEE Transactions on Multimedia, IEEE Transactions on Vehicular Technology, IEEE Transactions on Network and Service Management, IEEE Transactions on Cloud Computing, IEEE Transactions on Industrial Informatics, IEEE Transactions on Industrial Electronics and IEEE Transactions on Big Data), academic books and international conferences.