Received December 17, 2020, accepted January 10, 2021, date of publication January 18, 2021, date of current version February 1, 2021.
Digital Object Identifier 10.1109/ACCESS.2021.3051945

A Secure and Privacy-Preserving Machine Learning Model Sharing Scheme for Edge-Enabled IoT

XIANFEI ZHOU[1], KAI XU[1], NAIYU WANG[2], (Graduate Student Member, IEEE), JIANLIN JIAO[1], NING DONG[1], MENG HAN[1], AND HAO XU[1]
[1]State Grid Beijing Electric Power Company, Beijing 100031, China
[2]School of Control and Computer Engineering, North China Electric Power University, Beijing 102206, China
Corresponding author: Naiyu Wang (shininess_y@163.com)

This work was supported by the Science and Technology Project of State Grid Corporation of China under Grant 520223190047.

ABSTRACT With the popular use of IoT devices, edge computing has been widely applied in the Internet of things (IoT) and regarded as a promising solution for its wide distribution, decentralization, low latency. At the same time, in response to the massive computing data and intelligent requirements of various applications in the IoT, artificial intelligence (AI) technology has also achieved rapid development. As a result, edge intelligence (EI) for the Internet of Things has attracted widespread attention. Driven by the requirement that making full use of data, machine learning (ML) models trained in EI are usually shared. However, there may be some security and privacy issues due to the openness and heterogeneity of edge intelligence. How to ensure flexible data access and data security as well as the accountability for edge nodes and users in EI model sharing have become important issues. In this article, we propose a Ciphertext Policy Attribute Based Proxy Re-encryption (CP-ABPRE) scheme with accountability to address the security and privacy issues in EI model sharing. In our scheme, a user can delegate the access right to others to make model access more flexible. Furthermore, each entity that may need to be held accountable is embedded a unique ID to achieve traceability. Finally, security analysis and performance evaluation are given to prove that our scheme is CPA secure and does not lose much efficiency with more features.

INDEX TERMS Internet of Things, edge intelligence, model sharing, CP-ABPRE, accountability.

I. INTRODUCTION

Cisco's report [1] has predicted that in the future, a lot of IoT data will be generated from the edge side. If these huge data are processed by cloud computing, the process of sending them to the cloud will consume a lot of bandwidth resource and bring great computing pressure to the cloud [2]. Additionally, the high latency of cloud computing is not suitable for the tasks that require real-time response. Driven by this trend, computing power is already shifting from the centralized cloud to edge side, or data source side [3], [4]. At the same time, AI technology can be used to quickly perceive and train local data, which can not only adapt to the rapidly changing environment, such as real-time prediction, but also reduce delay and greatly improve computing efficiency [2], [5]. Therefore, edge intelligence (EI) – the integration of edge computing and AI will undoubtedly form a strong driving force for the development of applications in the Internet of Things. However, one of the key challenges of EI model training is the issue of data privacy [6]. On the one hand, the sensitivity of private data prevents source data and even trained models from being freely share, on the other hand, the openness and heterogeneity of edge devices also put forward higher requirements for flexible access control and security [7]. Moreover, because of the difficulty of managing heterogeneous edge nodes and the potential laziness and dishonest behaviors of the edge servers, an efficient accountability mechanism is necessary [8], [9].

The Ciphertext Policy Attribute-Based Encryption (CP-ABE) scheme [10], [11] can achieve the flexible data access. The party that wants to share the model develop access policies to decide which users/servers can access

The associate editor coordinating the review of this manuscript and approving it for publication was Zhenyu Zhou.

the model, then encrypt the data under the access policy. Model requesters who have the access right can request the encrypted data and decrypt the model successfully. But this has a limitation, for example, when a superior user is unable to view the uploaded data in time, he/she needs to delegate his/her access right to other trusted but not authorized subordinate users to ensure that the model parameters is processed in time, while ensuring that the superior user 's private key is not leaked [12].

To address the problem mentioned above, we can combine the Proxy Re-encryption (PRE) technology [13], [14] with CP-ABE. The combination of the two enables the translation of the ciphertext encrypted under an access policy into the ciphertext encrypted under another access structure [15], [16]. However, few researchers have considered the possible effects of edge server laziness or dishonest behavior. There are massive edge nodes in the IoT and the distribution area of edge nodes is wide. Therefore, it is necessary to introduce accountability and consistency verification mechanisms in the access control system to track the malicious behavior of edge nodes. There are also some ABE schemes that can provide traceability or audit, such as [17], [18]. Some researchers also work to verify the consistency of the data when the processing of the data is entrusted to a third-party agency [19], [20]. But these works did not implement proxy re-encryption which is needed in the distributed edge computing which requires low latency and flexible data access. In this article, we combine CP-ABE and PRE in the edge intelligence scene for the IoT, realizing security model sharing and access control between edge nodes. In particular, we use edge nodes as proxy servers to take the re-encryption computation load to reduce re-encryption latency and provide responsibility tracking to constrain the behavior of edge nodes.

The main contributions of this article can be summarized in three aspects:

1) Our proposed scheme guarantees the security, access control and proxy re-encryption of ML model parameters sharing in EI which meets the demands of edge computing for data security and flexible access. Moreover, it addresses the problem that sometimes users must delegate the access right to other unauthorized users.

2) Our accountable scheme can distinguish the reason for decryption failures and traces back to the party that is responsible for the failure. Because of the complex security risks that cloud computing faced, it is essential to embed the appropriate accountability mechanism.

3) By integrating the public/secret key pair technique into the key generation phase, our scheme can efficiently prevent the key abuse problem.

II. RELATED WORK

A. EDGE INTELLIGENCE

Edge intelligence is a key driver for the development of the IoT as its computing is close to the underlying data sources end, enabling low-latency and low-cost data processing [21]. Xiao *et al.* [22] designed a federal edge intelligence framework that allows edge servers to evaluate the required sample data based on resource consumption in the IoT and their own data processing capabilities, which significantly improves resource utilization efficiency.

In addition to the framework design of EI applications, there are also some studies on data security in EI models. Some studies ensure system security by detecting malicious behavior. Xu *et al.* [23] proposed a data-driven robust network anomaly detection for network security, and trained a model capable of detecting and identifying network anomalies through a four-stage design. In order to prevent the privacy leakage caused by the vulnerability of IoT devices. Li *et al.* [24] designed a kernel level resource audit tool, KLRA, to collect resource-sensitive events and issue security warnings at a low cost. Some studies that use blockchain technology to design security frameworks [21], [25], [26]. Zhang *et al.* [21] proposed a resource scheduling scheme based on blockchain under the cross-domain sharing scenario, and designed the edge transaction approval mechanism. The experiment proves that the system can realize flexible and safe service and improve service ability.

Some research is devoted to protecting private data in the IoT. Du *et al.* [27] firstly stratified the edge IoT and studied the privacy protection of machine learning in data aggregation. Ma *et al.* [28] proposed a federal data cleaning protocol, Febclean, to meet the privacy protection requirements in the data cleaning phase of machine learning, which can realize data cleaning without compromising users' privacy in EI scenario. However, little research has been devoted to implementing access control of training models in the sharing process.

B. SECURITY AND PRIVACY-PRESERVATION

In order to make full use of data, there may be model sharing in EI. Ensuring data security and user privacy is a major area of interest within the field of EI. To further achieve fine-grained access control and flexible data access, CP-ABE and PRE technology have been widely used to solve the security and privacy issues [29]–[32]. Some researchers also combine CP-ABE with PRE to achieve more functions. Liang *et al.* [33] first proposed a new CP-ABPRE scheme that is proven CCA secure and they further improve the security level of the ABPRE in [34]. Yu *et al.* [35] exploited PRE to delegate many computation tasks to the untrusted cloud servers. Lin *et al.* introduced verifiability for AB-PRE to guarantee the consistency of the data before and after re-encryption by proxy in [36]. However, most existing works focus on the security level, regardless the potential laziness and dishonest behaviors of the semi-trusted third party.

The edge computing has been adopted in many applications in IoT. Due to the potential laziness and dishonest behaviors of edge servers, accountability for edge servers needs to be considered. The former accountable methods [17], [18], [37] usually focus on tracing a specific entity, they always ignore the accountability of some malicious users.

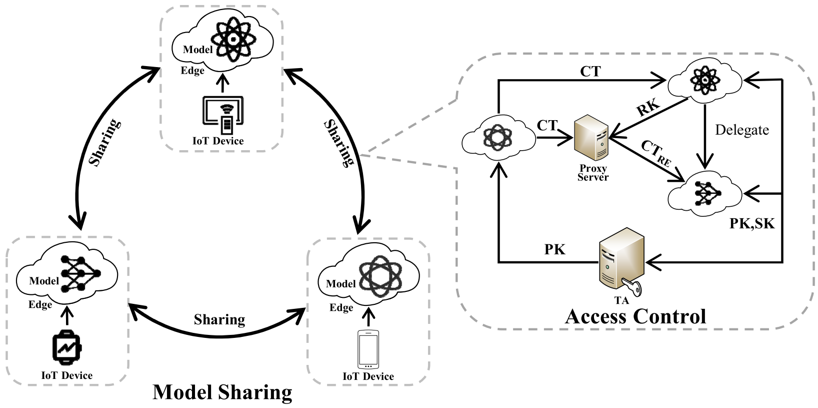

FIGURE 1. System model.

III. PRELIMINARIES

In this section, we give some preliminary that will be used in the scheme.

A. BILINEAR MAPS

Let the group $\mathbb{BG} =< \mathbb{G}_0, \mathbb{G}_1, p, g, e >$ denotes the bilinear pairing group, $\mathbb{G}_0$ and $\mathbb{G}_1$ are multiplicative cyclic groups of prime order p, g is the generator of $\mathbb{G}_0$. $e : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$ denotes the bilinear map. For all $a, b \in \mathbb{Z}_p$ and $u, v, w \in \mathbb{G}_0$, it has

1) $e(u^a, v^b) = e(u, v)^{ab}$.
2) $e(g, g) \neq 1$.
3) $e(uv, w) = e(u, w) \cdot e(v, w)$.

B. LINEAR SECRET SHARING SCHEMES (LSSS)

We employ LSSS into our proposed scheme to construct the access structure.

A secret sharing scheme $\prod$ over the set $\mathbb{Z}_p$ is called linear if:

1) The shares for each party form a vector over $\mathbb{Z}_p$.

2) Let $(\mathcal{M}, \rho)$ denotes the access structure under the LSSS. Let $\mathcal{M}$ be a $l \times n$ matrix, where l is the number of attributes in the set associated with the access structure and n is a variable defined by the LSSS turning method. ρ is a function which associates the rows of matrix $\mathcal{M}$ to attributes, where $\rho(i) \in \{Att_1, Att_2, ...Att_U\}$. In [38], they present how to share/recover the secret, and we will show the details of these procedures in the algorithms below.

C. PROXY RE-ENCRYPTION (PRE)

In our proposed scheme, PRE enables delegation for access rights by converting a ciphertext encrypted by the owner's public key into a ciphertext that can be decrypted by a semi-honest proxy private key. Suppose Alice, who wants to pass M to Bob. It can be described as follows:

1) $PRE.Setup(par) \rightarrow (pk_a, sk_a), (pk_b, sk_b)$: it takes system parameter par as inputs outputs the public key and the private key of the Alice and Bob.

2) $PRE.Enc(par, M, pk_a) \rightarrow C_{pk_a} = Enc(pk_a, M)$: it takes par, M and Alice's public key as inputs and outputs the ciphertext.

3) $PRE.ReKeyGen(par, sk_a, pk_b) \rightarrow K_{a\rightarrow b}$: it takes par, Bob's public key and Alice's secret key as inputs and outputs the Re-encryption key.

4) $PRE.ReEnc(par, K_{a\rightarrow b}, C_{a(n)}) \rightarrow C_{b(n+1)}$: it takes par, Re-encryption key and the n-th re-encrypted ciphertext as inputs and outputs the n +1-th re-encrypted ciphertext that can be decrypted by Bob.

5) $PRE.Dec(par, sk_b, C_{b(n+1)}) \rightarrow M$: it takes par, re-encryption ciphertext and Bob's secret key as inputs and outputs M.

TABLE 1. The description of symbols.

Symbol	DEFINITION
PK	Public Key
MK	Master Key
SK	Secret Key for users
RK	Re-encryption key
upk,usk	A unique public and secret key pair for each user
TA	Trusted Authority
M	Plaintext model parameters
CT	Ciphertext
CT_{RE}	Re-encrypted ciphertext
ID_{ct}, ID_p	Identity of CT and edge nodes

IV. MODELS OF OUR SYSTEM

A. OVERVIEW

Fig.1 shows the proposed system model architecture, including EI model sharing and a brief flow of access control during the sharing process. The definition of some symbols is listed in Table 1. To meet the need to make good use of data, parties (we use users below instead) may share the model parameters trained by each edge node. Considering security and privacy issues, we added access control and proxy re-encryption in model sharing. The Trusted Authority (TA) is a trusted party and the parties are semi-trusted, it follows the proposed protocols in general, however, it also attempts to dig out as much information as possible. We employ edge node as the proxy server to perform the re-encryption of users' ciphertexts (CT). Some users may collude with others for the purpose of obtaining illegal access right, but none of them is willing to reveal their personal key pairs, preventing others from stealing their own secrets. The detailed description is as follows:

1) The role of TA is the generator the public key (*PK*) and the master key (*MK*). New users will request registration from TA, each legitimate user will be assigned a unique identity and a pair of keys (usk, upk). Model requesters submit the set of attributes S to TA, and get the corresponding secret key (*SK*).

2) Model owner encrypts the model parameters under an access structure *AS* to get the ciphertext (*CT*).

3) We set Alice and Bob as the model requesters, they can get the requested ciphertext from the edge node, and if their attributes satisfy the access structure, the *CT* can be decrypted successfully.

4) If Alice wants to delegate her access right to Bob who does not have access to the corresponding model parameters, she first generates a re-encryption key (RK) according to her own secret key and a specified access structure AS^* and sends it to the proxy server (the edge node). Note that the original access structure *AS* and the new one AS^* are totally disjoint and the attributes of Bob must satisfy the access structure AS^*.

5) The re-encryption will be performed by the proxy server with the re-encryption key to get the re-encrypted ciphertexts (CT_{RE}).

6) Bob can request the CT_{RE}, and if Alice has the access right for the *CT*, Bob can decrypt the CT_{RE} successfully even if he does not have the appropriate permissions for the *CT*.

7) Trusted Authority records the identities of those who have decrypted the re-encrypted ciphertexts and the edge nodes who have performed the re-encryption, and traces any malicious users or edge nodes.

B. ALGORITHM DESCRIPTION

The following algorithms are included in our proposed scheme.

Setup(1^λ) $\rightarrow$(PK, MK, (usk, upk)). This algorithm is responsible for generating system parameters, the *PK* and the *MK* as well as the pair of keys (usk, upk) of each legitimate user.

Key_Gen(S, PK, MK, upk)$\rightarrow$$SK$. This algorithm generates a secret key SK for each user through attributes S.

Data_Encryption(M, PK, $(\mathcal{M}, \rho)$) $\rightarrow$$CT$, ID_{ct}. This algorithm is responsible for generating the ciphertext *CT* and the identity ID_{ct} of *CT*.

Dec_CT(ID_{req}, CT, SK, PK, usk)$\rightarrow$ M or $\perp$. This algorithm is responsible for decrypting the *CT* to get the plaintext model parameters M if the decryption is successful or $\perp$ if fails.

Re_Key_Gen(SK, usk, PK, ID_{ct}, ID_p, $(\mathcal{M}', \rho')$) $\rightarrow$$RK$. This algorithm is responsible for generating the re-encryption key *RK*. *RK* can be used for transforming *CT* to CT_{RE} which is under a new access structure $(\mathcal{M}', \rho')$. Additionally, the attributes set embedded in *SK* cannot satisfy the $(\mathcal{M}', \rho')$ since $(\mathcal{M}', \rho')$ and $(\mathcal{M}, \rho)$ are disjoint.

Re_Enc(ID_{ct}, CT, PK, RK)$\rightarrow$$CT_{RE}$ or $\perp$. This algorithm is responsible for generating a re-encrypted ciphertext CT_{RE} under a new access structure. If re-encryption fails, it outputs.

Dec_RCT(ID_{ct}, CT_{RE}, SK, PK, usk)$\rightarrow$ M or $\perp$. This algorithm is responsible for decrypting the CT_{RE} to get the plaintext model parameters M if the decryption is successful or $\perp$ if fails.

Check(ID_{ct}, ID_p, CT_{RE}, SK, PK, usk)$\rightarrow$*True* or *False*. This algorithm is used for checking if there is a need of accountability for the edge nodes.

Trace(SK, γ, MK, PK)$\rightarrow$$Uid$ or $\perp$. This algorithm outputs the corresponding user id *Uid* to denote the SK is valid. Or it outputs $\perp$ to denote that the *SK* is invalid.

C. SECURITY MODEL

Let $\mathcal{A}$ be the adversary to attack our CP-ABPRE scheme and $\mathcal{C}$ be the challenger. The game is described as below.

Init: $\mathcal{A}$ specifies the challenge access structure AS^* for the game, all the challenge ciphertext are encrypted under AS^*.

Setup: The algorithm is run by Challenger $\mathcal{C}$, which gives the public parameters PK to the adversary $\mathcal{A}$ and keeps MK to itself.

Phase 1: This phase contains several steps:

(1) $\mathcal{A}$ issues queries for private keys corresponding to the sets of attributes $S_1, \cdots, S_{q_1}$. (q_1 are integers randomly chosen by $\mathcal{A}$). $\mathcal{C}$ return the secret keys for $\mathcal{A}$ by calling ***Key_Gen***(Σ, PK, MK, upk). Note that if any set S issued by $\mathcal{A}$ satisfies AS^*, then aborts.

(2) On inputs SK, usk, and a re-encryption access structure A^*, $\mathcal{C}$ returns ***Re_Key_Gen***(SK, usk, PK, A^*) $\rightarrow$$RK$ to $\mathcal{A}$.

(3) On inputs ID_{ct}, RK and a ciphertext *CT* under access policy AS^*, C runs ***Re_Enc***(ID_{ct}, CT, RK, PK) and returns the result to A.

Challenge: $\mathcal{A}$ generates two messages of the same length, M_0 and M_1, and sends them to $\mathcal{C}$. $\mathcal{C}$ randomly flips a coin $b \in \{0, 1\}$, and encrypts M_b under AS^*. Then $\mathcal{C}$ calls ***Data_Encryption***(M_b, PK, AS^*)$\rightarrow$$CT^*$, and the generated ciphertext CT^* is given to $\mathcal{A}$.

Phase 2: $\mathcal{A}$ continues to issue his queries to $\mathcal{C}$ as in **Phase 1**.

Guess: $\mathcal{A}$ outputs its guess $b' \in \{0, 1\}$ for b and wins the game if $b' = b$.

The advantage of $\mathcal{A}$ in this game is defined as

$$Adv(\mathcal{A}) = \left|\Pr\ [\text{b'=b}] - {}^1/_2\right|$$

Definition 2: Our CP-ABPRE scheme is chosen plaintext attack (CPA) secure if no Probabilistic Polynomial Time (PPT) adversary $\mathcal{A}$ can win the following game with non-negligible advantage.

V. DESCRIPTION OF OUR SYSTEM

In this section, we give the algorithm details of basic scheme, the proxy re-encryption and accountability, as well as the security analysis.

A. BASIC SCHEME

1) SETUP(1^λ) → PK, MK

TA runs this algorithm. First, it chooses a bilinear group $\mathbb{BG}$ and U random elements: $h_1, \ldots, h_U \in \mathbb{G}_0$. Each of the elements is associated with an attribute in the system. The algorithm additionally chooses $\alpha, a, b \in \mathbb{Z}_p$ and then introduces two collision-resistant hash functions $H_1 : \{0,1\}^* \to \mathbb{Z}_p, H_2 : \{0,1\}^* \to \mathbb{G}_0$. The public key and the master key are published as

$$PK = \left\{g, e(g,g)^\alpha, g^b, g^a, h_1 \cdots h_U, H_1, H_2\right\} \quad (1)$$

$$MK = \{a, \alpha, b\} \quad (2)$$

Additionally, TA is also responsible for users' registration. Each legitimate user will be assigned a unique identity $Uid \in \mathbb{Z}_p$ and a pair of keys (usk, upk). Note that $usk = k \in \mathbb{Z}_p, upk = g^k$, where usk is chosen at random. TA will generate a certificate that includes the user's upk, and send it along with the corresponding usk to the user.

The PK will be public to the system and the MK will be kept secret by TA.

2) KEY_GEN(S, PK, MK, UPK) → SK

Each user obtains his/her attribute set and secret key only if he/she registers to TA and is verified as legal. For illegal users, TA will reject the key generation request. Otherwise, TA will assign the attribute set and generate the SK for him/her. First, user submits the set of attributes S and $upk = g^k$ to TA to request the SK. After receiving the user's request, TA first chooses random exponents $t \in \mathbb{Z}_p$ and $\beta \in \mathbb{Z}_p$ Then it outputs the corresponding SK as

$$\{K = g^{k\alpha} g^{at+b}, K' = g^{k\alpha} g^{(at+b)k},\ L' = g^{\alpha k+\beta}$$
$$L = g^t, \forall x \in S, K_x = h_x^t\} \quad (3)$$

3) DATA_ENCRYPTION(M, PK, $(\mathcal{M}, \rho)$) → CT

Model owner encrypts the data under the access structure $(\mathcal{M}, \rho)$ as defined in section II. At the beginning of encryption, model owner selects a vector $\vec{v} = (s, y_2, y_3, \ldots, y_n) \in \mathbb{Z}_p^n$ at random, where $y_2, y_3, \ldots, y_n$ are used for sharing the encryption secret s. For $i = 1, \cdots, l,$, it calculates $\lambda_i = \mathcal{M}_i \cdot \vec{v}$, where the vector $\mathcal{M}_i$ denotes the i-th row of $\mathbb{M}$. Then it selects a string c and $r_1, r_2, \ldots, r_l \in \mathbb{Z}_p$ at random, then computes $ID_{ct} = c$, $C = Me(g,g)^{\alpha s}$, $C' = g^s$ and $\forall 1 \le i \le l, C_i = g^{a\lambda_i} h_{\rho(i)}^{-r_i}, \hat{D}_i = g^{(H_1(ID_{ct})+b)\lambda_i}, D_i = g^{r_i}$.

Finally, CT is published as

$$CT :< ID_{ct}, C, C', \forall 1 \le i \le l, \{C_i, \hat{D}_i, D_i\} > \quad (4)$$

4) DEC_CT(ID_{req}, CT, SK, PK, USK) → M OR ⊥

There are two kinds of decryption algorithms in this article. One is to decrypt the original ciphertext, while the other is to decrypt the re-encrypted ciphertext. Both are performed by model requesters. We first introduce the former one.

Model requester can request the ciphertext according to the ID_{req} which denotes the ID of the ciphertext requested by the model requester. However, only when his/her attributes set S satisfies the $(\mathcal{M}, \rho)$ that corresponding to the CT, can he/she successfully decrypt the data. Let $I \subset \{1, 2, \ldots, l\}$ be defined as $I = \{i : \rho(i) \in S\}$. If S doesn't satisfy the $(\mathcal{M}, \rho)$, then it aborts. Otherwise, the algorithm can find a set $\{\omega_i | i \in I\}$ such that the following holds $\sum_{i \in I} \omega_j \lambda_j = s$. Then it computes:

$$F = \frac{e\left(C', K \cdot g^{H_1(IDreq)}\right)}{\prod_{i \in S} \left(e(L, C_i)\, e(D_i, K_x)\, e\left(g, \hat{D}_i\right)\right)^{\omega_i}}$$
$$= \frac{e(g^s, g^{k\alpha} g^{at+b} g^{H_1(IDreq)})}{\prod_{i \in S} \left(e\left(g^t, g^{a\lambda_i} h_{\rho(i)}^{-r_i}\right) e\left(g^{r_i}, h_x^t\right) e(g,g)^{(H_1(IDct)+b)\lambda_i}\right)^{\omega_i}}$$
$$= \frac{e(g,g)^{k\alpha s}\, e(g,g)^{ats}\, e(g,g)^{(b+H_1(IDreq))s}}{e(g,g)^{ats}\, e(g,g)^{(H_1(IDct)+b)s}}$$
$$= e(g,g)^{k\alpha s}. \quad (5)$$

The ciphertext be decrypted only when ID_{req} and ID_{ct} are equal.

Then it recovers the data as

$$C \Big/ (F)^{1/k} = M \cdot e(g,g)^{\alpha s} \Big/ e(g,g)^{\alpha s} = M_{.(6)} \quad (6)$$

Otherwise, it outputs ⊥ to denote a decryption failure.

B. PROXY RE-ENCRYPTION (PRE)

In this section, we give the details on how to realize PRE based on the basic scheme. Each edge node that performs proxy re-encryption is distributed with an identity ID_p.

1) RE_KEY_GEN(SK, USK, PK, ID_{ct}, IDP, $(\mathcal{M}', \rho')$) → RK

Suppose that a model requester named Alice wants to delegate her access right to Bob who does not have access to the corresponding model parameters. Let $(\mathcal{M}', \rho')$ denote the new access policy specified by Alice. S_{Alice} denote Alice's attributes set. She will run the following algorithm with her own decryption key SK and private key usk.

The algorithm first chooses a random $\varepsilon \in \mathbb{Z}_p$, then computes $K_{rk} = (K)^{\varepsilon/k} = g^{\alpha\varepsilon} g^{\varepsilon(at+b)/k}, L_{rk} = L^{\varepsilon/k} = g^{\varepsilon t/k}, L'_{rk} = g^{\varepsilon/k}, x \in S_{Alice}, K_{rk,x} = K_x^{\varepsilon/k} = h_x^{\varepsilon t/k}$. The rk_1 is denoted as

$$rk_1 =< K_{rk}, L_{rk}, L'_{rk}, x \in S_{Alice}, K_{rk,x} > \quad (7)$$

With the new access policy $(\mathcal{M}', \rho')$, this algorithm randomly selects a vector $\vec{v}' = (s', y'_2, y'_3, \ldots, y'_n) \in \mathbb{Z}_p^n$. For $i = 1, \cdots, l,$, it calculates $\lambda'_i = \mathcal{M}'_i \cdot \vec{v}'$. It also selects $r'_1, r'_2, \ldots, r'_l \in \mathbb{Z}_p$ and $R_0, R_1 \in \mathbb{Z}_p$, then computes

$$J = \varepsilon \cdot e(g,g)^{\alpha s'},$$
$$J' = g^{s'},$$
$$J_i = g^{a\lambda'_i} h_{\rho(i)}^{-r'_i},$$

$$J_i' = g^{r_i'},$$
$$\hat{J}_i = g^{(H_1(ID_{ct})+b)\lambda_i'},$$
$$\tilde{J} = (R_0||R_1)e(g,g)^{(\alpha+b)s'},$$
$$V = H_2(ID_{ct})^{R_0} H_2(ID_p)^{R_1}.$$

The rk_2 is denoted as

$$rk_2 = \left\langle V, J, J', \forall 1 \le i \le l, J_i, J_i', \hat{J}_i \right\rangle. \tag{8}$$

Finally, the re-encryption key is set as $RK = \{rk_1, rk_2\}$.

2) RE_ENC(ID_{ct}, CT, RK, PK) $\rightarrow$ CT_{RE} OR $\perp$

The edge node runs this algorithm below. The edge node checks if S_{Alice} satisfies the access structure $(\mathcal{M}, \rho)$, if not, it output $\perp$; Otherwise, it chooses constants ω_j such that the following holds $\sum_{i \in I} \omega_j \lambda_j = s$. Then it computes

$$\begin{aligned} A &= \prod_{i \in I} \left(e\left(L_{rk}, C_i\right) e\left(D_i, K_{rk,x}\right) e\left(L_{rk}', \hat{D}_i\right) \right)^{\omega_i} \\ &= \prod_{i \in I} \left(e\left(g,g\right)^{at\varepsilon\lambda_i/k} e\left(g,g\right)^{\lambda_i(b+H_1(ID_{ct}))\varepsilon/k} \right)^{\omega_i} \\ &= e\left(g,g\right)^{ats\varepsilon/k} e\left(g,g\right)^{(b+H_1(ID_{ct}))s\varepsilon/k}, \end{aligned} \tag{9}$$

And it continues to compute as follows

$$\begin{aligned} F' &= \frac{e\left(C', K_{rk} \cdot \left(L_{rk}'\right)^{H_1(ID_{ct})}\right)}{A} \\ &= \frac{e(g^s, g^{\alpha\varepsilon} g^{(at+b)\varepsilon/k} g^{H_1(ID_{ct})\varepsilon/k})}{A} \\ &= \frac{e\left(g,g\right)^{\alpha s\varepsilon} e\left(g,g\right)^{(at+b+H_1(ID_{ct}))s\varepsilon/k}}{e\left(g,g\right)^{ats\varepsilon/k} e\left(g,g\right)^{(b+H_1(ID_{ct}))s\varepsilon/k}} \\ &= e\left(g,g\right)^{\alpha s\varepsilon}. \end{aligned} \tag{10}$$

Let $C_{re,1} = F', C_{re,2} = rk_2, C_{re,3} = C$ and let CT_{RE} denote the re-encrypted ciphertexts, then it will be published as

$$CT_{RE} =< C_{re,1}, C_{re,2}, C_{re,3} > . \tag{11}$$

3) DEC_RCT(ID_{ct}, CT_{RE}, SK, PK, USK) $\rightarrow$ M OR $\perp$

We now introduce the second decryption algorithm. The execution of this algorithm is similar to *Dec_CT*. Bob run this algorithm, he decrypts CT_{RE} as follows:

$$\begin{aligned} A &= \prod_{i \in S} \left(e\left(L, J_i\right) e\left(J_i', K_x\right) e\left(\hat{J}_i, g\right) \right)^{\omega_i} \\ &= \prod_{i \in S} \left(e\left(g^t, g^{a\lambda_i'} h_{\rho(i)}^{-r_i'}\right) e\left(g^{r_i'}, h_x^t\right) e\left(g,g\right)^{(b+H_1(ID_{ct}))\lambda_i'} \right)^{\omega_i} \\ &= \prod_{i \in S} \left(e\left(g,g\right)^{at\lambda_i'} e\left(g,g\right)^{(b+H_1(ID_{ct}))\lambda_i'} \right)^{\omega_i} \\ &= e\left(g,g\right)^{ats'} e\left(g,g\right)^{(b+H_1(ID_{ct}))s'} \end{aligned} \tag{12}$$

$$\begin{aligned} F &= e\left(J', K \cdot g^{H_1(ID_{ct})}\right) \Big/ A \\ &= \frac{e(g^{s'}, g^{k\alpha} g^{at+b} g^{H_1(ID_{ct})})}{A} \\ &= e\left(g,g\right)^{k\alpha s'} \end{aligned} \tag{13}$$

The data is recovered by computing

$$J \Big/ F^{1/usk} = \varepsilon \cdot e(g,g)^{\alpha s'} \Big/ e\left(g,g\right)^{k\alpha s'/k} \varepsilon, \tag{14}$$

$$= C_{re,3} \Big/ (C_{re,1})^{1/\varepsilon} = M. \tag{15}$$

Otherwise, it outputs $\perp$ to denote a decryption failure.

C. ACCOUNTABILITY

In this section, we show how to extend our scheme with the property of accountability. It focuses on two entities: edge server and users.

For edge nodes: We design a new form of ciphertext embedding a unique identity. Additionally, the identity will be broadcasted to its target user group. To successfully decrypt the ciphertext, the identity of the decrypter must match with the embedded identity.

However, once the decryption fails, we cannot distinguish the reasons. Hence, it is reasonable that accountability must be provided. This process can be represented by the following subroutine:

1) CHECK(ID_{ct}, ID_p, CT_{RE}, SK, PK, USK)$\rightarrow$ TRUE OR FALSE

The execution of the algorithm is similar to *Dec_CT*. It decrypts as follows

$$\begin{aligned} A' &= \prod_{i \in S} \left(e\left(L^{1/usk}, J_i\right) e\left(J_i', K_x\right) \right)^{\omega_i} \\ &= \prod_{i \in S} \left(e\left(g^t, g^{a\lambda_i'} h_{\rho(i)}^{-r_i'}\right) e\left(g^{r_i'}, h_x^t\right) \right)^{\omega_i} \\ &= \prod_{i \in S} \left(e\left(g,g\right)^{at\lambda_i'} \right)^{\omega_i} \\ &= e\left(g,g\right)^{ats'} \end{aligned} \tag{16}$$

$$\begin{aligned} F' &= \frac{e\left(J', K'\right)}{A'} \\ &= \frac{e(g^{s'}, g^{k\alpha} g^{at+bk})}{A'} \\ &= \frac{e\left(g,g\right)^{k\alpha s'} e\left(g,g\right)^{ats'} e\left(g,g\right)^{bks'}}{e\left(g,g\right)^{ats'}} \\ &= e\left(g,g\right)^{k(\alpha+b)s'} \end{aligned} \tag{17}$$

Then it sets $\mathcal{F} = F^{1/usk} = e\left(g,g\right)^{(\alpha+b)s'}$.

Recover the two random numbers and set a new parameter V^*:

$$R_0||R_1 = \tilde{J} \Big/ \mathcal{F} = (R_0||R_1)e(g,g)^{(\alpha+b)s'} \Big/ e(g,g)^{(\alpha+b)s'}. \tag{18}$$

$$V^* = H_2(ID_{ct})^{R_0} H_2(ID_p)^{R_1}. \tag{19}$$

Check it with V in CT_{RE}, if they are equal and *Dec_RCT*(ID_{ct}, CT_{RE}, *SK, PK, usk*)$\rightarrow$ $\perp$, the output is *False*, which means it is necessary to initiate investigation to edge nodes. Otherwise, the output is *True*, which means an unmatched attribute set leads to the decryption failure.

For users: Each time TA traces a suspicious model requester, it asks he/she to securely submit his/her *SK* and a parameter γ. The user will sign the public parameter with his/her own secret key usk, which is computed as $\varphi = (g^a)^{1/k}$, $\gamma = \varphi \cdot e(g,g)^{\alpha k}$.

2) TRACE(SK, γ, MK, PK)→ UID OR ⊥.

This algorithm first parses SK into several key components. Then it recovers the user's upk with MK.

$$(L'/g^{\beta})^{1/\alpha} = (g^{\alpha k+\beta}/g^{\beta})^{1/\alpha} = g^k \tag{20}$$

This upk will be considered as an index to search its corresponding identity Uid^*. TA obtains φ by computing $\varphi = \gamma/e(g^{\alpha}, g^k) = g^{a/k}$.

Next, this algorithm will verify whether the SK can pass all the following checks.

3) KEY SANITY CHECK

$$e(\varphi, upk) = e(g,g)^a \tag{21}$$

$$e(K,g) = e(upk, g^{\alpha})e(\varphi, L)e(g, g^b) \tag{22}$$

$$\forall x \in S, e(L, h_x) = e(K_x, upk) \tag{23}$$

Only when (21), (22) and (23) hold, can SK be viewed as a well-formed key. If the Uid^* is proved to be valid, the algorithm outputs Uid. Otherwise, it outputs ⊥.

Note that we embed a unique id in the ciphertexts. Each time when there is a model requester (regardless of he/she is a suspicious user or not) requesting for this ciphertext, it will be convenient to record who has taken the ciphertext.

D. SECURITY ANALYSIS

Theorem 1: Suppose that the construction of [11] is CPA secure, then our basic scheme is CPA secure.

Proof: Our scheme is based on the scheme in [11] that is proved to be CPA secure under the decisional q-BDHE assumption. We take the scheme in [11] as scheme Γ. Similarly, we can also build a simulator $\mathbb{B}$ to attack the scheme Γ. Suppose an adversary $\mathcal{A}$ can attack our basic scheme with non-negligible advantage.

Init: $\mathcal{A}$ gives a challenge structure $(\mathcal{M}^*, \rho^*)$, where $\mathcal{M}^*$ has n^* columns.

Setup: $\mathbb{B}$ randomly chooses a parameter $\alpha' \in \mathbb{Z}_p$, setting $\alpha = \alpha' + a^{q+1}$ by letting $e(g,g)^{\alpha} = e(g^a, g^{a^q})e(g,g)^{\alpha'}$. The master parameter can be canceled out during the decryption. Hence, the simulator simply set it as *b'*.

Each x $(1 \leq x \leq U)$ begins when chooses a random value z_x. Let X denote the set of indices i and set $\rho^*(i) = x$. The group elements are created as follows:

$$h_x = g^{z_x} \prod_{i \in X} g^{a\mathcal{M}^*_{i,1}/b_i} \cdot g^{a^2\mathcal{M}^*_{i,2}/b_i} \ldots g^{a^{n^*}\mathcal{M}^*_{i,n^*}/b_i}.$$

Phase 1: $\mathbb{B}$ receives key queries for a set S which does not satisfy $(\mathcal{M}^*, \rho^*)$. It chooses a random parameter $r \in \mathbb{Z}_p$. Then it finds a vector $\vec{\omega} = (\omega_1, \omega_2, \ldots, \omega_{n^*}) \in \mathbb{Z}_p^{n^*}$ such that $\omega_1 = -1$ and for all i where $\rho^*(i) \in S$ we have that $\vec{\omega} \cdot \mathcal{M}^*_i = 0$. Then $\mathbb{B}$defines t by computing

$$r + \omega_1 a^q + \omega_1 a^{q-1} + \cdots + \omega_{n^*} a^{q-n^*+1}.$$

It sets

$$L = (g^k)^r \prod_{i=1,\ldots,n^*} (g^{a^{q+1-i}})^{\omega_i} = g^{kt}.$$

$$K = (g^k)^{\alpha'+b'} g^{ar} \prod_{i=2,\ldots,n^*} (g^{a^{q+2-i}})^{\omega_i}.$$

Suppose that if there is no attribute in S involved in the challenge structure, we can simply let $K_x = L^{z_x}$. Otherwise, let X denote the set of attributes involved in the structure, and $\mathbb{B}$ computes as follows

$$K_x = L^{z_x} \prod_{i \in X} \prod_{j=1,\ldots,n^*} (g^{(a^j/b_i)r} \prod_{\substack{k=1,\ldots n^* \\ k \neq j}} (g^{a^{q+1+j-k}/b_i})^{\omega_k})^{\mathcal{M}^*_{i,j}}.$$

Challenge: $\mathcal{A}$ chooses two messages M_0 and M_1, and submits them to $\mathbb{B}$. $\mathbb{B}$ randomly flips a coin b, and computes

$$C = M_b T \cdot e(g^s, g^{\alpha'}), C' = g^s.$$

To create the element C_i , randomly chooses y'_i and the vector is computed as follows

$$\vec{v} = (s, sa + y'_2, sa^2 y'_3, \ldots, sa^{n-1} + y'_{n^*}) \in \mathbb{Z}_p^{n^*}.$$

Then it chooses $r'_1, r'_2 \ldots r'_\ell$ at random. For $i = 1, \ldots, n^*$, it computes

$$D_i = g^{-r'_i} g^{-sb_i}$$

$$C_i = h^{r'_i}_{\rho^*(i)} (\prod_{j=2,\ldots,n^*} (g^a)^{\mathcal{M}^*_{i,j} y'_j})(g^{b_i \cdot s})^{-z_{\rho^*(i)}} \cdot (\prod_{k \in R_i} \prod_{j=1,\ldots,n^*} (g^{a^j \cdot s \cdot (b_i/b_k)})^{\mathcal{M}^*_{k,j}}),$$

$$\hat{D}_i = (\prod_{j=2,\ldots,n^*} (g^{(H_1(ID_{ct})+b')})^{\mathcal{M}^*_{i,j} y'_j})(g^{b_i \cdot s})^{-z_{\rho^*(i)}} \cdot (\prod_{k \in R_i} \prod_{j=1,\ldots,n^*} (g^{a^j \cdot s \cdot (b_i/b_k)})^{\mathcal{M}^*_{k,j}}).$$

Phase 2: Repeat *Phase 1*.

Guess: $\mathcal{A}$ outputs its guess on $b \in \{0,1\}$. If $b' = b$, $\mathbb{B}$ outputs 0 to guess that $T = e(g,g)^{a^{q+1}s}$. Else, it outputs 1. If T is a tuple, the simulator gives a perfect simulation so that we can have

$$\Pr\left[\mathcal{B}(\mathbf{y}, T = e(g,g)^{a^{q+1}s}) = 0\right] = 1/2 + \mathrm{Adv}_{\mathcal{A}}$$

Obviously, if $\mathcal{A}$ can attack our scheme with a non-negligible advantage, we can build a simulator $\mathbb{B}$ that attack the scheme Γ with a non-negligible advantage.

Theorem 2: Our scheme is collusion resistant.

Proof: Our scheme is proved to be collusion resistant. There is a random parameter t being inserted in each of the key components. No one can get access to this random number.

Additionally, at the beginning of the system set up, each of the users will be assigned a pair of keys (usk, upk). Each upk corresponds to a unique user id. When the procedure of key generation and key distribution are performed, some of the components are encrypted with user's upk. A user can decrypt successfully only if he/she owns the usk, which is known only by SK's true owner.

VI. PERFORMANCE EVALUATION

A. EXPERIMENT SETTINGS

Our experiments are implemented in the python 3.6 environment on the top of the Charm-Crypto-0.43 framework [39]. We choose the SS512 curve as the pairing curve. All experiments are tested on a laptop with 2.5 GHz Intel Core i5 processor and 8GB RAM and on a virtual machine with ubuntu 18.04.3.

B. NUMERICAL ANALYSIS

We analyze the our scheme from the numerical point of view in this section. The average time spent on various operations on group Γ_0 and Γ_1 is given in Table 2. The elements being tested are randomly chosen from the corresponding groups. We can notice that the multiplication takes a short time compared to the other operations. The exponentiation in group Γ_0 takes much more time than it in group Γ_1. Pairing is the slowest operation compared to the other operations. We set x as the number of attributes of the user who applied for the secret key, y as the number of attributes involved in the access structure.

TABLE 2. The average time spent on various operations (Milliseconds).

Groups	Mul	Exp	Paring
G_0	0.009	2.56	3.16
G_1	0.008	0.23	

In key generation phase, it takes 4+x exponentiation operations in Γ_0 to generate the secret key. In re-encryption key generation phase, because the re-encryption key is generated for a specific ciphertext, the key must contain both the information of the ciphertext and the *information* of the new access structure. Therefore, the overhead of re-encryption key generation of PRE is much more expensive than the key generation of ABE, that is 1 multiplication operation, 7+x +y exponentiation operations in Γ_0, 2 multiplication operations, 2 exponentiation operations in Γ_1. But the encryption overhead of PRE is relatively small, the details is given below.

In encryption phase, it takes 1+4y exponentiation operations in Γ_0 and 1 multiplication operation, 1 exponentiation operation in Γ_1 to encrypt the data. In re-encryption phase, it takes 1 multiplication operation and 1 exponentiation operation in Γ_0, 2y multiplication operations and y exponentiation operations in Γ_1, 4 paring operations.

In decryption phase, it takes 1 multiplication operation and 1 exponentiation operation in Γ_0, 1+2y multiplication operations, 1+y exponentiation operations in Γ_1, and 1+3y paring operations. And the decryption overhead of PRE is only one more multiplication operation and exponentiation operation in Γ_1 than it of ABE.

C. EXPERIMENT ANALYSIS

We give the comparison of our scheme and waters' scheme [11] on which our scheme is based. The number of attributes on the abscissa refers to the number of attributes of the user who applied for the secret key in Fig. 2 and the number of attributes included in the access structure in Fig. 3. And it refers to the number of attributes involved in decryption in Fig. 4, we set the relationship of all the attributes included in the access structure to "AND" gate to represent the number of attributes involved in decryption. It is worth noting that the re-encrypted ciphertext is re-encrypted under a new access structure which is satisfied by the attributes of delegated user. Therefore, the number of attributes in the encryption and decryption phase of the PRE scheme is the number of

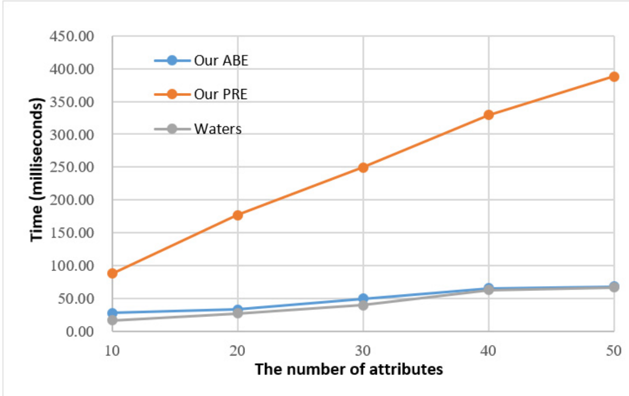

FIGURE 2. The key generation costs.

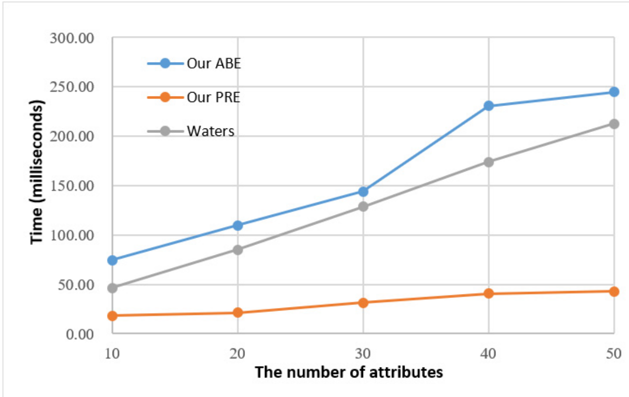

FIGURE 3. The encryption costs.

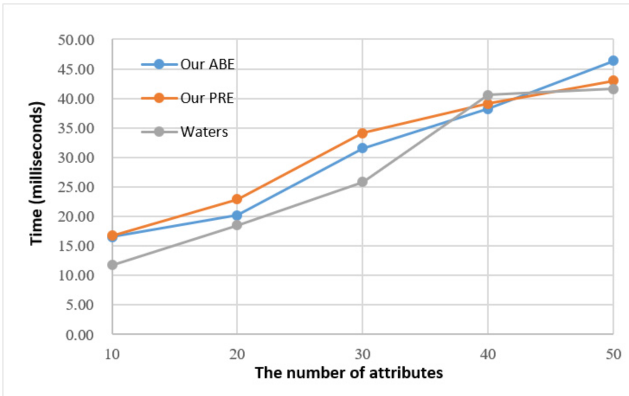

FIGURE 4. The decryption costs.

attributes included in the new access structure. Moreover, all the time is in milliseconds. We set "Our ABE" to denote our basic ABE scheme and "Our PRE" to denote our proxy re-encryption scheme. "Waters" denotes the scheme in [11]. All data are tested 100 times and averaged.

The key generation phase is performed by TA. We can conclude from Fig.2 that with the number of attributes increases, the computational cost in our ABE scheme and Waters' scheme has a smaller increasing trend and is almost the same with our scheme implementing more features such as accountability and verification. Due to the inherent characteristics of proxy re-encryption, its computational cost is larger than that of ABE during the key generation phase.

As shown in Fig. 3, the encryption phase of our ABE scheme and Waters' scheme is performed by the data owner with our PRE scheme being performed by the edge node. To achieve accountability and verification in our scheme, we must add several exponential operations in the encryption phase, it makes the computational cost of our ABE scheme larger than that of Waters' scheme. But our PRE scheme has a very small computation overhead, which guarantees low latency for edge nodes to process data.

As shown in Fig. 4, the decryption phase is performed by data requester. In PRE scheme, Alice delegates her access right to Bob who has no access right. Bob uses his own secret key to decrypt the re-encrypted ciphertext like a normal decryption of our ABE scheme. Therefore, the decryption costs of PRE are related to the number of attributes involved in decryption, just like our ABE scheme. The decryption computation overhead is almost the same in these three schemes with our scheme implementing both PRE and ABE.

VII. CONCLUSION

In this article, we construct the CP-ABPRE scheme with accountability to address the data security and privacy issues in EI model sharing. It provides several techniques to achieve access control as well as the proxy re-encryption to protect its underlying plaintext model parameters in model sharing. Additionally, users can judge the behavior of the edge nodes in our scheme using a reasonable accountability checking mechanism. By integrating the public/secret key pair technique into the key generation, our proposed scheme can efficiently prevent the key abuse problem and is able to identify the key users in the investigation of a decryption failure. The analysis proves that our scheme can satisfy the security requirements, and the proposed scheme can effectively defend against both individual and colluded malicious users. The performance evaluation proves that our scheme has not significantly reduced efficiency after implementing accountability and proxy re-encryption in CP-ABE scheme. In our future work, we will consider adding incentives on the basis of this system to encourage more edge nodes to share their models actively. Moreover, we will also look into establishing secret key update and cancellation mechanisms to make the key management more flexible.

REFERENCES

[1] Cisco, "Cisco global cloud index: Forecast and methodology, 2018–2023," White Paper. Accessed: Mar. 2020. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.pdf
[2] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, and J. Zhang, "Edge intelligence: Paving the last mile of artificial intelligence with edge computing," *Proc. IEEE*, vol. 107, no. 8, pp. 1738–1762, Aug. 2019, doi: 10.1109/JPROC.2019.2918951.
[3] S. Liao, J. Li, J. Wu, W. Yang, and Z. Guan, "Fog-enabled vehicle as a service for computing geographical migration in smart cities," *IEEE Access*, vol. 7, pp. 8726–8736, 2019.
[4] X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, and M. Chen, "In-edge AI: Intelligentizing mobile edge computing, caching and communication by federated learning," *IEEE Netw.*, vol. 33, no. 5, pp. 156–165, Sep. 2019.
[5] S. Deng, H. Zhao, W. Fang, J. Yin, S. Dustdar, and A. Y. Zomaya, "Edge intelligence: The confluence of edge computing and artificial intelligence," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 7457–7469, Aug. 2020, doi: 10.1109/JIOT.2020.2984887.
[6] H. Liao, Y. Mu, Z. Zhou, M. Sun, Z. Wang, and C. Pan, "Blockchain and learning-based secure and intelligent task offloading for vehicular fog computing," *IEEE Trans. Intell. Transp. Syst.*, early access, Jul. 21, 2020, doi: 10.1109/TITS.2020.3007770.
[7] Y. Li, S. Yao, R. Zhang, and C. Yang, "Analyzing host security using D-S evidence theory and multisource information fusion," *Int. J. Intell. Syst.*, vol. 36, no. 2, pp. 1053–1068, Feb. 2021, doi: 10.1002/int.22330.
[8] M. Caprolu, R. Di Pietro, F. Lombardi, and S. Raponi, "Edge computing perspectives: Architectures, technologies, and open security issues," in *Proc. IEEE Int. Conf. Edge Comput. (EDGE)*, Milan, Italy, Jul. 2019, pp. 116–123.
[9] J. Ren, G. Yu, Y. He, and G. Y. Li, "Collaborative cloud and edge computing for latency minimization," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 5031–5044, May 2019.
[10] Q. Zhang, Y. Gan, L. Liu, X. Wang, X. Luo, and Y. Li, "An authenticated asymmetric group key agreement based on attribute encryption," *J. Netw. Comput. Appl.*, vol. 123, pp. 1–10, Dec. 2018.
[11] B. Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization," in *proc. Int. Workshop Public Key Cryptogr.*, vol. 6571, 2011, pp. 53–70.
[12] Z. Guan, X. Liu, L. Wu, J. Wu, R. Xu, J. Zhang, and Y. Li, "Cross-lingual multi-keyword rank search with semantic extension over encrypted data," *Inf. Sci.*, vol. 514, pp. 523–540, Apr. 2020.
[13] M. Mambo and E. Okamoto, "Proxy cryptosystems: Delegation of the power to decrypt ciphertexts," *IEICE Trans. Fundamentals Electron., Commun. Comput. Sci.*, vol. 80, pp. 54–63, Jan. 1997.
[14] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved proxy re-encryption schemes with applications to secure distributed storage," *ACM Trans. Inf. Syst. Secur.*, vol. 9, no. 1, pp. 1–30, Feb. 2006.
[15] X. Liang, Z. Cao, H. Lin, and J. Shao, "Attribute based proxy re-encryption with delegating capabilities," in *Proc. 4th Int. Symp. Inf., Comput., Commun. Secur. (ASIACCS)*, 2009, pp. 276–286.
[16] S. Luo, J. Hu, and Z. Chen, "Ciphertext policy attribute-based proxy re-encryption," in *Proc. Int. Conf. Inf. Commun. Secur.*, 2010, pp. 401–415.
[17] J. Ning, Z. Cao, X. Dong, and L. Wei, "White-box traceable CP-ABE for cloud storage service: How to catch people leaking their access credentials effectively," *IEEE Trans. Dependable Secure Comput.*, vol. 15, no. 5, pp. 883–897, Sep. 2018.
[18] Z. Liu, S. Duan, P. Zhou, and B. Wang, "Traceable-then-revocable ciphertext-policy attribute-based encryption scheme," *Future Gener. Comput. Syst.*, vol. 93, pp. 903–913, Apr. 2019.
[19] J. Li, Z. Guan, X. Du, Z. Zhang, and J. Wu, "An efficient encryption scheme with verifiable outsourced decryption in mobile cloud computing," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–6.
[20] Z. Guan, X. Lu, W. Yang, L. Wu, N. Wang, and Z. Zhang, "Achieving efficient and privacy-preserving energy trading based on blockchain and ABE in smart grid," *J. Parallel Distrib. Comput.*, vol. 147, pp. 34–45, Jan. 2021.
[21] K. Zhang, Y. Zhu, S. Maharjan, and Y. Zhang, "Edge intelligence and blockchain empowered 5G beyond for the industrial Internet of Things," *IEEE Netw.*, vol. 33, no. 5, pp. 12–19, Sep. 2019, doi: 10.1109/MNET.001.1800526.
[22] Y. Xiao, Y. Li, G. Shi, and H. V. Poor, "Optimizing resource-efficiency for federated edge intelligence in IoT networks," 2020, *arXiv:2011.12691*. [Online]. Available: http://arxiv.org/abs/2011.12691

[23] S. Xu, Y. Qian, and R. Q. Hu, "Data-driven edge intelligence for robust network anomaly detection," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 3, pp. 1481–1492, Jul. 2020, doi: 10.1109/TNSE.2019.2936466.

[24] D. Li, Z. Zhang, W. Liao, and Z. Xu, "KLRA: A kernel level resource auditing tool for IoT operating system security," in *Proc. IEEE/ACM Symp. Edge Comput. (SEC)*, Oct. 2018, pp. 427–432.

[25] Z. Zhou, B. Wang, M. Dong, and K. Ota, "Secure and efficient vehicle-to-grid energy trading in cyber physical systems: Integration of blockchain and edge computing," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 50, no. 1, pp. 43–57, Jan. 2020, doi: 10.1109/TSMC.2019.2896323.

[26] B. Cao, Y. Li, L. Zhang, L. Zhang, S. Mumtaz, Z. Zhou, and M. Peng, "When Internet of Things meets blockchain: Challenges in distributed consensus," *IEEE Netw.*, vol. 33, no. 6, pp. 133–139, Nov. 2019, doi: 10.1109/MNET.2019.1900002.

[27] M. Du, K. Wang, Y. Chen, X. Wang, and Y. Sun, "Big data privacy preserving in multi-access edge computing for heterogeneous Internet of Things," *IEEE Commun. Mag.*, vol. 56, no. 8, pp. 62–67, Aug. 2018.

[28] L. Ma, Q. Pei, L. Zhou, H. Zhu, L. Wang, and Y. Ji, "Federated data cleaning: Collaborative and privacy-preserving data cleaning for edge intelligence," *IEEE Internet Things J.*, early access, Sep. 30, 2020, doi: 10.1109/JIOT.2020.3027980.

[29] S.-Y. Tan, K.-W. Yeow, and S. O. Hwang, "Enhancement of a lightweight attribute-based encryption scheme for the Internet of Things," *IEEE Internet Things J.*, vol. 6, no. 4, pp. 6384–6395, Aug. 2019.

[30] K. Liang, M. H. Au, J. K. Liu, W. Susilo, D. S. Wong, G. Yang, T. V. X. Phuong, and Q. Xie, "A DFA-based functional proxy re-encryption scheme for secure public cloud data sharing," *IEEE Trans. Inf. Forensics Security*, vol. 9, no. 10, pp. 1667–1680, Oct. 2014.

[31] S. Maiti and S. Misra, "P2B: Privacy preserving identity-based broadcast proxy re-encryption," *IEEE Trans. Veh. Technol.*, vol. 69, no. 5, pp. 5610–5617, May 2020.

[32] Q. Zhang, X. Wang, J. Yuan, L. Liu, R. Wang, H. Huang, and Y. Li, "A hierarchical group key agreement protocol using orientable attributes for cloud computing," *Inf. Sci.*, vol. 480, pp. 55–69, Apr. 2019.

[33] K. Liang, L. Fang, W. Susilo, and D. S. Wong, "A ciphertext-policy attribute-based proxy re-encryption with chosen-ciphertext security," in *Proc. 5th Int. Conf. Intell. Netw. Collaborative Syst.*, Xi'an, China, Sep. 2013, pp. 552–559.

[34] K. Liang, H. A. Man, W. Susilo, D. S. Wong, and G. Yang, "An adaptively CCA-secure ciphertext-policy attribute-based proxy re-encryption for cloud data sharing," in *Proc. Int. Conf. Inf. Secur. Pract. Exper.*, 2014, pp. 448–461.

[35] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving secure, scalable, and fine-grained data access control in cloud computing," in *Proc. IEEE INFOCOM*, Mar. 2010, pp. 1–9.

[36] S. Lin, R. Zhang, and M. Wang, "Verifiable attribute-based proxy re-encryption for secure public cloud data sharing," *Secur. Commun. Netw.*, vol. 9, no. 12, pp. 1748–1758, Aug. 2016.

[37] Z. Guan, J. Li, Y. Zhang, R. Xu, Z. Wang, and T. Yang, "An efficient traceable access control scheme with reliable key delegation in mobile cloud computing," *EURASIP J. Wireless Commun. Netw.*, vol. 2016, no. 1, pp. 1–11, Sep. 2016.

[38] 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 Gener. Comput. Syst.*, vol. 52, pp. 95–108, Nov. 2015.

[39] J. A. Akinyele, C. Garman, I. Miers, M. W. Pagano, M. Rushanan, M. Green, and A. D. Rubin, "Charm: A framework for rapidly prototyping cryptosystems," *J. Cryptograph. Eng.*, vol. 3, no. 2, pp. 111–128, Mar. 2013.

XIANFEI ZHOU received the master's degree from China Agricultural University, in 2013. He is currently an Engineer with the Automation Department, State Grid Beijing Electric Power Company. His current research interests include cloud security and the IoT security.

KAI XU received the master's degree from CUMTB, in 2014. He is currently an Engineer with the Automation Department, State Grid Beijing Electric Power Company. His current research interests include smart grid security, cloud security, and network security.

NAIYU WANG (Graduate Student Member, IEEE) is currently pursuing the master's degree with the School of Control and Computer Engineering, North China Electric Power University. Her current research interests include blockchain and applied cryptography.

JIANLIN JIAO received the master's degree from China Agricultural University, in 2003. He is currently a Senior Engineer with the Power Dispatching Control Center, State Grid Beijing Electric Power Company. His current research interests include safe operation of power systems and network security.

NING DONG received the master's degree from the Beijing Institute of Technology, in 2002. He is currently a Professor Senior Engineer with the Automation Department, State Grid Beijing Electric Power Company. His current research interests include automation of power network dispatching and network security.

MENG HAN received the master's degree from BJTU, in 2016. He is currently a Senior Engineer with the Automation Department, State Grid Beijing Electric Power Company. His current research interests include smart grid security, cloud security, and network security.

HAO XU received the bachelor's degree from Sichuan University, in 2001. He is currently a Senior Engineer with the Automation Department, State Grid Beijing Electric Power Company. His current research interests include smart grid security, cloud security, and network security.