Achieving Fuzzy Matching Data Sharing for Secure Cloud-Edge Communication

Chuan Zhang[1], Mingyang Zhao[1], Yuhua Xu[2], Tong Wu[1,3,*], Yanwei Li [4,*], Liehuang Zhu[1], Haotian Wang[5]

[1] School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing 100081, China
[2] School of Computer Science and Technology, Beijing Institute of Technology, Beijing 100081, China
[3] Yangtze Delta Region Academy of Beijing Institute of Technology, Jiaxing 314019, China
[4] National Computer Network Emergency Response Technical Team/Coordination Center of China, Beijing 100029, China
[5] College of Arts and Science, University of Pennsylvania, Philadelphia 19104, USA
[*] The corresponding author, email: tongw@bit.edu.cn

Abstract: In this paper, we propose a novel fuzzy matching data sharing scheme named FADS for cloud-edge communications. FADS allows users to specify their access policies, and enables receivers to obtain the data transmitted by the senders if and only if the two sides meet their defined certain policies simultaneously. Specifically, we first formalize the definition and security models of fuzzy matching data sharing in cloud-edge environments. Then, we construct a concrete instantiation by pairing-based cryptosystem and the privacy-preserving set intersection on attribute sets from both sides to construct a concurrent matching over the policies. If the matching succeeds, the data can be decrypted. Otherwise, nothing will be revealed. In addition, FADS allows users to dynamically specify the policy for each time, which is an urgent demand in practice. A thorough security analysis demonstrates that FADS is of provable security under indistinguishable chosen ciphertext attack (IND-CCA) in random oracle model against probabilistic polynomial-time (PPT) adversary, and the desirable security properties of privacy and authenticity are achieved. Extensive experiments provide evidence that FADS is with acceptable efficiency.

Keywords: fuzzy-matching; privacy-preserving set intersection; cloud-edge communication; data sharing

Received: Jan. 11, 2022
Revised: Apr. 19, 2022
Editor: Wen Wu

I. INTRODUCTION

Cloud computing has been widely applied in various domains to help users, especially the resource-constraint end devices, to enjoy convenient and low-cost computing and storage services. However, with the explosive growth of end devices, it is quite difficult for the end devices to connect with the cloud servers with low response time. To deal with this issue, cloud-edge computing [1], as an emerging paradigm that exploits the computing, storage, and communication capacities of edge devices, has drawn significant attention [2]. The system architecture of cloud-edge computing is shown in Figure 1. By integrating the resources of both edge devices and cloud servers, cloud-edge computing offers a set of advantages such as providing a fast response for end devices, reducing bandwidth constraints, and relieving network congestion [3, 4]. Based on the report released by Grand View Research, the market of cloud-edge computing is expected to reach USD 61.14 billion by 2028 [5].

As an intermediate layer, the edge nodes handle the large scale of data transmission between the user and cloud, which however may raise severe privacy concerns. Firstly, although edge computing stores and proceeds data more close to the end devices compared with the cloud, the edge devices cannot be trusted. As the rising of cyber attacks, the edge devices are vulnerable to the attacks, such as eavesdropping, unauthorized modification, and unauthorized access to the

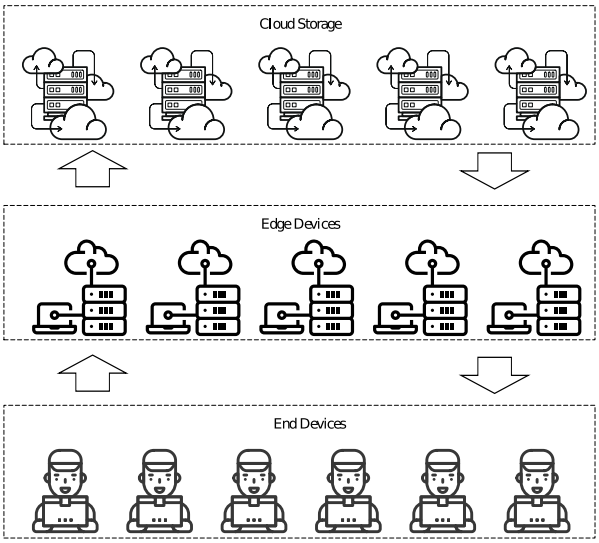

Figure 1. *System architecture of cloud-edge computing.*

system, etc., which causes that the users are deterred from sharing their data for concerns on the leakage of sensitive information. Secondly, the behavior of data transmission between two users may cause information leakage by launching inference attacks. The accurate sharing will expose the relationship among the users, helping the observers to clarify the social relationship among the users. For the aforementioned security and privacy concerns, we summarize the security requirements of cloud-edge computing as follows: 1) Data confidentiality: the data can be recovered if and only if the decryption succeeds; 2) User privacy: the adversary cannot determine the exact sender or receiver even if it observes the data transmission occurs; 3) Collusion resistance: when the cloud colludes with some edge devices, the cloud cannot decrypt the ciphertext correctly.

To construct a secure data sharing scheme in the cloud-edge computing environment, we may consider attribute-based encryption (ABE), access control, and access control encryption (ACE) [6, 7]. However, in these conventional cryptographic primitives, the access policies are specified by only one side, which is a one-to-many communication mode. Additionally, in ACE, a fully trusted third party needs to be always online to participate in data sharing and prevent attacks from malicious senders and receivers. To realize that both sender and receiver can specify access policy for the other, a solution is matchmaking encryption (ME) [8], which provides an accurate matching between attribute and access policy. More specifically, the matching is measured both by the sender's policy, sender's attributes, receiver's policy, and receiver's attributes. From the sender side, the sender's policy indicates the specific attributes of the receiver who can decrypt the ciphertext. From the receiver side, the receiver's policy indicates the specific attributes of the sender whose ciphertext can be decrypted. In addition, ME guarantees that if and only if the matching succeeds, the message will be recovered. Otherwise, nothing will be revealed. However, accurate matching may not satisfy the requirements of real-world applications, which require a many-to-many communication mode. Specifically, the many-to-many communication mode indicates that one sender can specify access policies for multiple receivers and vice versa. For example, in the healthcare system [9, 10], suppose that the hospital cooperates with other organizations to develop a novel treatment for some diseases (i.e., COVID-19). The hospital may only allow the organizations that meet its access policy to access the corresponding case data and the organization may only access the case data sent by hospitals that meet its access policy. With ME, the hospital needs to specify the corresponding access policy and generate the corresponding ciphertext for each organization, which causes a huge waste on computation and communication resources. Additionally, the aforementioned cryptographic primitives cannot support the receivers to dynamically specify the access policy for the senders during data sharing. Specifically, dynamic policies indicate that the receiver could specify the scope of the received message by changing the threshold value for the number of senders' attributes in the access policy.

To support the dynamic policies and many-to-many communication mode, we apply the ME with fuzzy matching. Specifically, compared with the aforementioned cryptographic primitives, ME with fuzzy matching has some advantages as follows:

- The message will be recovered if and only if the matching succeeds. Otherwise, nothing will be revealed except that the matching occurs.
- The policy is not specified by one side only, but both sides can make access policies for the opposite sides. Additionally, ME with fuzzy matching supports the receivers to dynamically specify the access policy for the senders during data sharing.

- The fuzzy matching allows a certain distance of error between the access policy and individual's attributes to realize the many-to-many communication mode in cloud-edge computing, to hide the accurate users involved in the sharing procedure.

In this paper, we introduce a novel fuzzy matching-based data sharing scheme, named FADS, for cloud-edge computing, derived from ME. Different from the conventional underlying cryptographic primitives, the decryption of fuzzy type ME is decided by both senders and receivers with error tolerance, forming a potential of many-to-many communication. Considering the aforementioned security requirements, FADS should be with the following characteristics: 1) Both sender and receiver can specify the access policies for establishing communication, and the receivers can dynamically specify the access policy for the senders during data sharing; 2) The matching should allow error tolerance, measured by the attributes and access policies; 3) The messages, attributes, and access policies of participants (senders and receivers) should stay secure even if the cloud colludes with some edge devices. 4) Some heavy computation will be taken by the edge devices to release the end devices from the burden of computation so that secure data sharing can be conducted among end devices efficiently by introducing cloud-edge computing.

1.1 Contribution

We formally define the notion of FADS in the cloud-edge computing environment, implemented by the pairing-based cryptosystem. We provide the security analysis and performance evaluation to FADS. Specifically, our contributions are listed as the following points:

1. We apply ME to present a pairing-based solution for constructing a fuzzy matching data sharing scheme named FADS. FADS is the first ME-based scheme supporting fuzzy matching by allowing the matching with error-tolerance between attributes and policies. If the decryption fails, nothing will be revealed, including the accurate attributes of users or why the matching fails.
2. We construct the encryption key for senders in the system with their own attributes. And we generate the decryption key for receivers with their own attributes. The key generation algorithm adopts Shamir's secret sharing [11] to accomplish error-tolerance in key components. Suppose that the subsets of the key components from opposite sides satisfying the access policies are in a designed offset. The message can be recovered from the ciphertext. Additionally, the receivers can dynamically specify the access policy for the senders during data sharing.
3. We prove the security of FADS to be with semantic security against any probabilistic polynomial-time adversary. Further, we prove two essential properties in FADS for cloud-edge computing, as privacy and authenticity.
4. We also evaluate FADS by conducting comparison experiments with some existing works to demonstrate that FADS is practical in real-world applications.

1.2 Literature Review

In this section, we compare our FADS with the existing works, shown as Table 1, in the perspective of security, privacy, authenticity, access control, and fuzzy matching.

Data Sharing for Cloud-edge Computing. Edge computing is a novel computing model that provides computation, storage, and networking services between end devices [25]. For supporting various data-driven services, a large scale of data is transmitted among the cloud and users in the network, including personal privacy and collaborated data. For instance, the static data and transmission process can be eavesdropped from the work logs by the malicious attacks in the whole procedure [26]. In cloud-edge computing, collaborated and personal data are treated as sensitive information, stored and processed near to users. In a cloud-edge computing system, the edge devices act as the proxy for end devices, which cannot protect the security and privacy of sensitive information for its constrained resource. In terms of our research on the state-of-the-art of cloud-edge computing, there are some existing works on resolving the security and privacy issues in cloud-edge computing. In [27], the authors figure out that secure edge devices usage is one of the crucial challenges in the cloud-edge computing environment. Hui et al. [28] suggested a secure data transmission scheme for edge computing, which relies on the synchronization of chaotic systems with different

Table 1. *Theoretical comparison with the existed data sharing schemes.*

Property	Security	Privacy	Authenticity	Access Control	Fuzzy Matching
[8]	√	√	√	bilateral	×
[12]	√	√	×	one-side	×
[13]	√	√	√	one-side	×
[14]	√	√	×	one-side	×
[15]	√	√	√	bilateral	×
[16]	√	√	√	bilateral	×
[17]	√	√	√	one-side	√
[18]	√	√	√	one-side	×
[19]	√	×	√	one-side	×
[20]	√	√	√	one-side	×
[21]	√	√	√	one-side	×
[22]	√	√	√	bilateral	×
[23]	√	√	√	one-side	×
[24]	√	√	√	one-side	×
FADS	√	√	√	bilateral	√

orders. The security of their proposed data transmission is based on the size of the keyspace. Their system is a one-to-one communication method. In [29], Xu et al. introduced a secure data transmission by using the physical layer with beamforming and artificial noise. With this method, the physical channel plays an essential role in ensuring system safety. From 2012, Gaurav [30] proposed the secure file transmission scheme, implemented by encryption. A sequence of encryption-based data sharing schemes has emerged for the distribution system [18, 12, 19, 13, 14, 20, 21]. In [12], Pan et al. suggested the ciphertext-policy attribute-based encryption (CP-ABE) to ensure the confidentiality of the information and share data among different domains, which organized by edge computing vehicles. In their scheme, the privacy of individuals is protected by using pseudo identities during the communication process. Liu et al. [13] constructed a secure data sharing scheme for mobile edge computing by applying the additional zero-knowledge proof (ZKP), secure multiparty computation, and succinct, transparent arguments of knowledge (STAK) to ensure the security and privacy of data, which is also a one-to-one communication model. Yang et al. [14] introduced a data sharing scheme by outsourcing the complex computation workloads of end-user devices to the edge nodes in a consortium blockchain system. The access control is realized by adopting the linear secret sharing scheme (LSSS). However, the data is encrypted with symmetric encryption, where the key management and distribution is a crucial issue to be solved.

To conclude the existing works, the data sharing schemes for edge computing are commonly in the form of one-to-one communication. To support the one-to-many communication, CP-ABE is considered to be an adequate methodology in practice. However, the access policy of CP-ABE is designed only by one side, so it cannot realize the many-to-many communication.

Matchmaking Encryption. The matchmaking encryption allows the sender and receiver to specify the access policy, simultaneously, firstly introduced by Ateniese et al. at CRYPTO'19 [8]. The subsequent works [15, 16, 22] make attempts to apply ME to fog computing to realize the fine-grained bilateral access control over outsourced data. Specifically, Chen et al. [16] avoids forging an identity in a conventional way and introduces certificateless matchmaking encryption (CL-ME) for the Internet of Things scenario. Danilo et al. [22] proposed an identity-based matchmaking encryption (IB-ME) scheme based on standard assumptions over bilinear groups. Our challenge in this work is to build fuzzy matching type ME, remaining privacy, authenticity, and security of the typical ME, to serve as a qualified building block of fuzzy matching data sharing scheme for cloud-edge computing. We take the privacy-preserving set intersection [31] into consideration to securely and privately compute the result of the access policies and attributes [32].

Linear Secret Sharing. In 1979, Shamir et al. [11] and Blakley et al. [33] introduced the linear secret sharing scheme. The process of the linear secret sharing scheme is as follows: 1) The sender splits the secret into multiple shares in an appropriate way; 2) The

sender distributes each share to different participants; 3) Participants cooperate to recover the secret. As one of the important tools of modern cryptography, the linear secret sharing scheme has been used in many practical applications [34–38]. Generally, the linear secret sharing scheme has two types, threshold secret sharing scheme and non-threshold secret sharing scheme. In non-threshold secret sharing scheme, a sequence of works [23, 24, 39] make attempts to apply the coding theory, linear universal hash functions, etc., to reach secret sharing. Specifically, Appala et al. [23] proposed a secret sharing scheme for compartmented access structure with lower bounds based on the Maximum Distance Separable (MDS) codes. Ronald et al. [24] constructed a linear secret sharing scheme based on linear code and linear universal hash functions in a black-box way. The non-threshold secret sharing scheme requires all receivers to participate in secret recovery, while the threshold secret sharing scheme only requires some members to participate in this procedure. Particularly, in fuzzy matching, the receiver only needs to meet some requirements specified by senders, which is similar to the requirement to recover the secret in the threshold secret sharing scheme. Therefore, we regard the threshold secret sharing scheme as a building block of our proposed scheme. The Lagrange interpolation polynomial method is typical to achieve the threshold secret sharing scheme. Specifically, the Lagrangian interpolation polynomial has the property that if the data is in the original data set used to generate the interpolation function, the corresponding Lagrangian coefficient equals 1. Otherwise, the Lagrangian coefficient equals 0. Based on this property, the Lagrangian interpolation polynomial method can be used to achieve secret recovery in threshold secret sharing. Thus far, a sequence of works [17, 40, 32, 41, 42] have been proposed to achieve the threshold secret sharing scheme. Specifically, Song et al. [40] proposed a secure fuzzy matching scheme based on symmetric-key threshold predicate encryption (STPE) and proxy re-encryption for vehicular crowdsourcing system. Their scheme achieves privacy-preserving threshold-based task matching and data transmission between worker and requester. Amit et al. [17] proposed a fuzzy identity-based encryption (IBE) scheme based on the Lagrange interpolation polynomial method. Their scheme is both error-tolerant and secure against collusion attacks. However, in these methods, the access policy is specified only by one side. To achieve that both sender and receiver can specify policy for the other, Giuseppe et al. [32] proposed a fuzzy secret handshake scheme based on the Lagrange interpolation polynomial method, and their scheme allows the handshakes to be based on bilateral matching. However, the aforementioned methods cannot support non-interactive bilateral matching in data sharing.

Thus, in this paper, we exploit the possibility of the Lagrange interpolation polynomial method for constructing efficient non-interactive bilateral fuzzy matching data sharing in the cloud-edge environment.

1.3 Organization

We organize our article as follows: in Section II, we introduce the mathematical preliminaries and cryptographic primitives used in our work. In SectionIII, we define the system model and architecture of FADS. In Section IV, we introduce the workflow of FADS and give a concrete construction. We formally analyze the security, privacy, and authenticity of our proposed scheme in Section V. In Section VI, we provide the theoretical comparison and experimental evaluation with the existing relevant works. In Section VII, we conclude our work and discuss our future works.

II. PRELIMINARY

In this section, we introduce some mathematical preliminaries and cryptographic primitives used in FADS.

2.1 Bilinear Group

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

suppose that $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ are bilinear groups with the same prime order p. g and h are generators of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively. The bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ holds properties as follows:

1. Bilinearity: For any $g \in \mathbb{G}_1$, $h \in \mathbb{G}_2$ and $a, b \in \mathbb{Z}_p^*$, $e(g^a, h^b) = e(g, h)^{ab}$.
2. Non-degeneration: For any $g \in \mathbb{G}_1$, $h \in \mathbb{G}_2$, $e(g, h) \neq 1_{\mathbb{G}_T}$, where $1_{\mathbb{G}_T}$ denotes the generator of $\mathbb{G}_T$.

3. Efficient computation: For any $g \in \mathbb{G}_1$, $h \in \mathbb{G}_2$, there exists an algorithm to compute $e(g,h)$ efficiently.

If $\mathbb{G}_1 = \mathbb{G}_2$ holds, we call it symmetric pairing. Otherwise, there are two types of asymmetric pairing, depending on the existence of isomorphism function from $\mathbb{G}_2$ to $\mathbb{G}_1$.

Next, we list the computationally intractable problem used in our proposed data sharing scheme.

Definition 2 (Decisional Bilinear Diffie-Hellman Problem (BDH)). *Given g_i^a, g_j^b, g_k^c and $e(g_1,g_2)^z$ determine whether $e(g_1,g_2)^{abc} = e(g_1,g_2)^z$, where $i,j,k \in \{1,2\}$. Given an algorithm to generate the group as $\mathcal{G}$, we define the following distribution*

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

Let $\mathcal{A}$ denote a PPT adversary breaking the BDH problem.

$$\begin{aligned}\mathrm{Adv}_{\mathcal{A}}^{\mathrm{BDH}}(\lambda) \overset{def}{=} \; &|\Pr[\mathcal{A}(D, e(g_1,g_2)^{abc})]\\ &-\Pr[\mathcal{A}(D, e(g_1,g_2)^{z})]|\end{aligned}$$

is negligible in the security parameter λ.

Definition 3 (Decisional Modified Bilinear Diffie-Hellman Problem (MBDH)). *Given g_i^a, g_j^b, g_k^c and $e(g_1,g_2)^z$ determine whether $e(g_1,g_2)^{\frac{ab}{c}} = e(g_1,g_2)^z$, where $i,j,k \in \{1,2\}$. Given an algorithm to generate the group as $\mathcal{G}$, we define the following distribution D as:*

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

Let $\mathcal{A}$ denote a PPT adversary breaking the MBDH problem.

$$\begin{aligned}\mathrm{Adv}_{\mathcal{A}}^{\mathrm{MBDH}}(\lambda) \overset{def}{=} \; &|\Pr[\mathcal{A}(D, e(g_1,g_2)^{\frac{ab}{c}})]\\ &-\Pr[\mathcal{A}(D, e(g_1,g_2)^{z})]|\end{aligned}$$

is negligible in the security parameter λ.

2.2 Lagrange Interpolation Polynomial

For a set of points $\{(x_j, y_j)\}$ for $y_j = f(x_j)$, the Lagrange interpolation polynomial $q(\cdot)$ is the unique algebraic polynomial of the lowest degree, coinciding with $f(\cdot)$, commonly used in secret sharing schemes. Given the number of points (x_j, y_j) satisfying $y_j = f(x_j)$ as n, we define the Lagrange coefficient

$$\triangle_{i,S}(x) = \prod_{j \in S, j \neq i} \frac{x - x_j}{x_i - x_j}.$$

To obtain $q(x)$, we compute the following equation:

$$q(x) = \sum_{i \in S} y_i \cdot \triangle_{i,S}(x).$$

2.3 Privacy-Preserving Set Intersection (PSI)

General two-party computation was firstly introduced by Yao [43], in which the players share their values and cooperatively evaluate the result. The recent work on privacy-preserving set operation [44] provides intersection, union, and mixed set operations. Here, we take PSI as a building block, contributing to securely computing decryption results. The details of this primitive consist of the following steps.

There are n participants $P_1, \cdots, P_n$. For $\forall P_i (1 \leq i \leq n)$, each has a private set $A_i = \{u_i^1, \cdots, u_i^l\}$, where $A_i \subset U$ with $U = \{u_1, \cdots, u_m\}$. Moreover, all parties know the encrypted vector $E(C) = (E(c_1), \cdots, E(c_m))$. If $u_j \in A$, that $A = A_1 \cup A_2 \cdots \cup A_m$, c_j is an even number. Otherwise, c_j is an odd number. The result of set intersection $T = \cap_{i=1}^n S_i$, where $S_i \in \{A_1, \cdots, A_n, \bar{A}_1, \cdots, \bar{A}_n\}$. $\bar{A}_i$ denotes the complement set of A_i to the set union. $E(\cdot)$ denotes a threshold ElGamal cryptosystem [44]. The security of PSI can be reduced to the security of the underlying cryptosystem.

III. DEFINITION

We introduce the system architecture of FADS for cloud-edge computing and formally define security models as IND-CCA security, privacy, and authenticity.

Privacy-preserving Set Intersection:

- *Input:* Private set $X_1, \cdots, X_n$ and a public encrypted vector $E(C) = (E(c_1), \cdots, E(c_m))$.
- *Output:* $E(B) = (E(b_1), \cdots, E(b_m))$. If $u_j \in T$, b_j is an even number; otherwise, b_j is an odd number.

Step 1. P_1 construct a vector in terms of two cases:

$$\begin{aligned} S_1 = A_1. \quad & u_j \in A_1, E(d_{1j}) = E(c_j) \\ & u_j \notin A_1, E(d_{1j}) = E(r_{1j}) \\ S_1 = \bar{A}_1. \quad & u_j \in A_1, E(d_{1j}) = E(r_{1j}) \\ & u_j \notin A_1, E(d_{1j}) = E(c_j) \end{aligned}$$

Note that r_{ij} is a randomly chosen odd number. P_1 sends $E(D_1) = (E(d_{11}), \cdots, E(d_{1m}))$ to P_2.

Step 2. For $2 \leq i \leq n-1$, P_i computes $Enc(D_i)$:

$$\begin{aligned} S_i = A_i. \quad & u_j \in A_i, E(d_{ij}) = E(d_{i-1j}) \\ & u_j \notin A_i, E(d_{ij}) = E(r_{ij}) \\ S_i = \bar{A}_i. \quad & u_j \in A_i, E(d_{ij}) = E(r_{ij}) \\ & u_j \notin A_i, E(d_{ij}) = E(d_{i-1j}) \end{aligned}$$

P_i sends $E(D_i) = (E(d_{i1}), \cdots, E(d_{im}))$ to P_{i+1}.

Step 3. As *Step 2*, P_n computes

$$E(D_n) = (E(d_{n1}), \cdots, E(d_{nm})),$$

and sets $E(B) = E(D_n)$.

3.1 System Architecture

We first define the system architecture of FADS for the cloud-edge computing environment. There are three layers in the system, the service layer, the intermediate layer, and the device layer. The responsibility of each layer is described as follows: 1) The service layer contains the cloud to store the data from users and the key generation center (KGC). As for KGC, it generates the key for users before joining the system. The sender's encryption key is computed with his/her own attributes. The receiver's decryption key is computed with its own attributes; 2) The edge devices compose the intermediate layer, responsible for the computation and storage between end devices and cloud; 3) The device layer consists of end devices

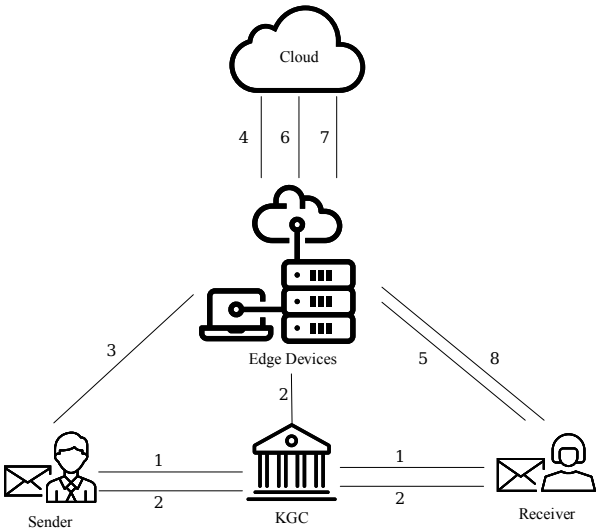

Figure 2. *System architecture of FADS for cloud-edge computing environment.*

for generating data. The sender specifies the access policy for the target receivers then encrypts the data by the encryption key and the sender's policy. The receiver also can specify the access policy to indicate who can send data to it. Then, decrypt the ciphertext by the decryption key and policy of the receiver. If the decryption successes, the receiver can recover the data from the ciphertext. Otherwise, it will not reveal anything, except the matching does not occur.

The system architecture is shown in Figure 2, and the notations used in this paper are listed in Table 2. At a high level, the process of FADS is described as follows: 1) Senders and receivers register to the KGC and obtain the corresponding secret key; 2) The KGC distributes system parameters to senders, receivers, and the edge devices; 3) Senders generate a series of ciphertexts and then send them to the edge devices; 4) The edge devices retain the ciphertexts used in the match phase and then upload the rest to the cloud; 5) Receivers send the ciphertexts of attributes and the ciphertexts of access policies used in PSI to the edge devices; 6) The edge devices execute matching and send the data request to the cloud; 7) The cloud returns the corresponding data to the edge devices; 8) the edge devices return the data to receivers, and receivers execute the decryption phase to recover the message.

More specifically, FADS consists of polynomial-time algorithms: Setup, SKGen, RKGen, Enc,

Table 2. *Notations used in FADS.*

Notation	Meaning
mpk	master public key
msk	master secret key
σ	attributes of sender
ρ	attributes of receiver
$\mathtt{ek}_\sigma$	encryption key of sender
$\mathtt{dk}_\rho$	decryption key of receiver
$\mathbb{R}$	policy of sender
$\mathbb{S}$	policy of receiver
$\mathcal{M}$	message space
C	ciphertext
d	degree of polynomial
q_1, q_2	$d-1$ degree polynomial
$\Delta_{i,\mathbb{S}_j}$	Lagrange coefficient of σ
$\Delta_{i,\mathbb{R}_j}$	Lagrange coefficient of ρ
$E(\cdot)$	encryption function of PSI
$H(\cdot)$	hash function
Θ	attribute set
$\mathcal{A}$	symbol of adversary
$\mathcal{B}$	symbol of simulator

Matching, and Dec.

- Setup. The system server takes the security parameter λ as the input. It outputs the master public key mpk and the master secret key msk.
- SKGen. The system server takes the master secret key msk and attributes $\sigma \in \{0,1\}^*$. It outputs an encryption key $\mathtt{ek}_\sigma$ for the sender.
- RKGen. The system server takes the master secret key msk and attributes $\rho \in \{0,1\}^*$. It outputs a decryption key $\mathtt{dk}_\rho$ for the receiver.
- Enc. The sender takes the encryption key $\mathtt{ek}_\sigma$, policy of sender: $\mathbb{R} : \{0,1\}^* \rightarrow \{0,1\}$, and a message $m \in \mathcal{M}$. It outputs a ciphertext C.
- Matching. The edge device takes the encrypted attributes of sender/receiver and the encrypted access policy of sender/receiver. It outputs *accepted* if the matching succeeds. Otherwise, *rejected*.
- Dec. The receiver takes the secret decryption key $\mathtt{dk}_\rho$, policy of receiver: $\mathbb{S} : \{0,1\}^* \rightarrow \{0,1\}$, and a ciphertext C. It outputs a message m or $\perp$.

Informally, when a ciphertext on m is generated honestly under the sender's encryption key and access policy of sender $\mathbb{R}$, the output of the decryption algorithm is conducted under the receiver's decryption key and access policy of receiver $\mathbb{S}$. m can be recovered from the ciphertext, if and only if ρ matches $\mathbb{R}$, and σ matches $\mathbb{S}$, simultaneously.

Definition 4 (Correctness)**.** *A fuzzy matching data sharing scheme with massage space $\mathcal{M}$ is correct if the security parameter $\lambda \in \mathbb{N}$, $\forall m \in \mathcal{M}$ to be encrypted, the sender's attributes and receiver's attributes $\sigma, \rho \in \{0,1\}^*$, the access policy of sender and receiver $\forall \mathbb{R}, \mathbb{S} : \{0,1\}^* \rightarrow \{0,1\}$, and $\forall(\mathtt{mpk}, \mathtt{msk})$ generated by* Setup(λ)*:*

$$\Pr[\mathsf{Dec}(\mathtt{dk}_\rho, \mathbb{S}, \mathsf{Enc}(\mathtt{ek}_\sigma, \mathbb{R}, m)) = m] \geq 1 - \epsilon,$$

if and only if $Dist(\mathbb{S}, \sigma) \geq d \wedge Dist(\mathbb{R}, \rho) \geq d$, and otherwise,

$$\Pr[\mathsf{Dec}(\mathtt{dk}_\rho, \mathbb{S}, \mathsf{Enc}(\mathtt{ek}_\sigma, \mathbb{R}, m)) = \perp] \geq 1 - \epsilon,$$

where $\mathtt{ek}_\sigma \leftarrow \mathsf{SKGen}(\mathtt{msk}, \sigma)$, $\mathtt{dk}_\rho \leftarrow \mathsf{RKGen}(\mathtt{msk}, \rho)$.

$Dist$ indicates the distance between the attributes and policies of the sender/receiver.

3.2 Threat Model

In our scheme, the KGC is a fully trusted third party, and all communications with the KGC are secure. Except for the KGC, each entity in our scheme can be an adversary. It is noted that we assume that the cloud can collude with the edge devices. Based on the capabilities of an adversary, we summarize the adversary into five types, i.e., malicious participant, cloud-only adversary, edge-only adversary, cloud-edge collusion adversary, and external adversary. Specifically, we describe the threat model of FADS as follows:

- **Malicious participant.** The participant (sender and receiver) possesses his/her own attributes and access policies and can access the ciphertexts generated by others. The malicious participant launches chosen ciphertext attack to obtain messages, attributes, and access policies of others.
- **Cloud-only adversary.** The cloud is responsible for storing the ciphertexts generated by senders. It is honest to perform the execution, but it launches ciphertext-only attack to obtain messages, attributes, and access policies of participants.

- **Edge-only adversary.** The edge device is responsible for executing the matching by PSI. The edge device can access the ciphertexts stored on the cloud, the ciphertexts of attributes, and the ciphertexts of access policies used in PSI, stored on the edge devices. The edge device launches ciphertext-only attack to obtain messages, attributes, and access policies of participants.
- **Cloud-edge collusion adversary.** The adversary can access the ciphertexts stored on the cloud, the ciphertexts of attributes, and the ciphertexts of access policies used in PSI, stored on the edge devices. Then, the adversary launches ciphertext-only attack to obtain messages, attributes, and access policies of participants.
- **External adversary.** External adversary can access the ciphertexts by eavesdropping on the communication channel between participants and the edge devices. The adversary launches ciphertext-only attack to obtain messages, attributes, and access policies of participants.

3.3 Security on FADS

The security characteristics of our proposed data sharing are formally defined in this part in terms of security, privacy, and authenticity. Informally, the security is on top of the computational indistinguishability between $\mathsf{Enc}(\mathtt{ek}_{\sigma_0}, \mathbb{R}_0, m_0)$ and $\mathsf{Enc}(\mathtt{ek}_{\sigma_1}, \mathbb{R}_1, m_1)$ with querying SKGen and RKGen. More specifically, we define our proposed data sharing is IND-CCA secure. The oracles $\mathcal{O}_S, \mathcal{O}_R$ are represented to SKGen and RKGen.

- **Setup**. The challenger runs Setup algorithm and publishes the public parameters.
- **Phase 1**. The challenger will allow the adversary to request the encryption keys and decryption keys from $\mathcal{O}_S, \mathcal{O}_R$, respectively. The adversary gives σ to $\mathcal{O}_S$ for getting ek_σ. And it provides ρ to $\mathcal{O}_R$ for getting dk_ρ.
- **Challenge**. The adversary chooses the sender's encryption key, the target receiver and two messages m_0, m_1, where $|m_0| = |m_1|$. The challenger encrypts $m_b, b \in \{0, 1\}$, by flipping a random coin. The challenger will send the ciphertext to the adversary. Note that the target receiver's decryption key cannot be requested in Phase 1.
- **Phase 2**. It is similar to *Phase 1*, except that the adversary cannot request the target receiver's decryption key.
- **Guess**. At the end of the game, the adversary outputs a guess b' on b.

We say the adversary wins the game if b' given by the adversary in the *Guess* phase is equal to b chosen by the challenger. Therefore, FADS is secure that

$$\mathrm{Adv}_{\mathcal{A}}^{\mathrm{IND-CCA}}[b' = b | \mathcal{A}^{\mathcal{O}_1, \mathcal{O}_2}(m_0, m_1, C^*)] \leq \epsilon$$

where ϵ is negligible. From [8], the security of ME implies privacy and authenticity. Thus, the security of FADS implies privacy and authenticity, as well.

3.4 Privacy on FADS

To protect the privacy of attribute, FADS should first guarantee that the adversary cannot recover the attributes from the ciphertext, encrypted by σ_0 or σ_1. Secondly, FADS should guarantee that if the matching fails, the adversary cannot know whose attributes cannot meet the other's policy.

To protect the privacy of access policy, FADS should first guarantee that the adversary cannot recover the access policy from the ciphertext, encrypted by $\mathbb{R}_0$ or $\mathbb{R}_1$. Secondly, FADS should guarantee that if the matching fails, the adversary cannot know whose access policy is not met.

In summary, we say the adversary breaks the privacy if $b' = b$ that the adversary gives the guess on b correctly. Therefore, FADS holds the privacy that

$$\mathrm{Adv}_{\mathcal{A}}^{\mathrm{Pri}}[b' = b | \mathcal{A}^{\mathcal{O}_1, \mathcal{O}_2}(ek_0, \mathbb{R}_0, ek_1, \mathbb{R}_1, C^*)] \leq \epsilon$$

where ϵ is negligible.

3.5 Authenticity on FADS

Authenticity demands that if a sender attempts to create a valid ciphertext with attributes σ, the sender must obtain the encryption key ek_σ from the KGC and use ek_σ to generate the ciphertext. Otherwise, the ciphertext will be invalid. The authenticity ensures that if a ciphertext can be decrypted correctly, it must be produced by an authenticated sender. More specifically, the sender encrypts m^* under a forged secret key ek^*

to generate C^*. Then, the receiver attempts to recover the message from C^*. However, the probability that the message can be recovered from C^* is negligible. Thus, FADS holds the authenticity that

$$\Pr_{\mathcal{A}}^{\text{Auth}}[m = \mathsf{Dec}(C^*)|\mathcal{A}(\sigma^*, C^*, ek^*, m^*)] \leq \epsilon$$

where $m^* \in \mathcal{M}$, ek^* was not queried before, and ϵ is negligible.

IV. FUZZY MATCHING DATA SHARING FOR CLOUD-EDGE COMPUTING

In this section, we introduce the workflow of FADS and the concrete construction of FADS for cloud-edge computing in terms of the aforementioned system architecture.

4.1 Workflow

The workflow of our proposed data sharing is shown as Figure 3. The working entities in the system are divided into KGC, End Devices, Edge Devices, and Cloud. Our system has four phases: **System Initialization**, **Registration**, **Data Synchronization**, and **Data Sharing**.

- **System Initialization**: Given a security parameter, KGC outputs the master secret key and the public parameters. Then, KGC distributes the public parameters to all legal end devices and edge devices.
- **Registration**: An end device can access the system after registering in KGC. This process can be divided into two cases: sender key generation and receiver key generation.
    1. **Sender Key Generation**: When an end device, as the sender, sends the attributes, KGC takes the master secret key and attributes to generate the encryption key. KGC sends the encryption key to the end device for encryption during data sharing.
    2. **Receiver Key Generation**: When an end device, as the receiver, sends the attributes, KGC takes the master secret key and attributes to generate the decryption key. KGC sends the decryption key to the end device for decryption during data sharing.
- **Data Synchronization**: In our system, the edge devices establish a ciphertext pool, an attribute pool, and an access policy pool and store the data in the corresponding pool. In addition, the edge devices will upload ciphertexts that have not been accessed for a long time to the cloud, and delete the data locally, thereby improving the resource utilization of edge devices.
- **Data Sharing**: In our system, the end devices, edge devices, and cloud collaborate the data sharing. The data sharing stage can be divided into three procedures: data encryption, matching, and data decryption.
    1. **Data Encryption**: An end device as the sender is aiming to send the data to a group of receivers whose attributes satisfy the access policy designed by the sender itself with a certain distance of error. The encrypted data is computed with the access policy and the encryption key. Then, it sends the ciphertext to the edge device.
    2. **Matching**: The edge devices conduct the matching procedure by applying the PSI over the attributes of end devices and their access policies. The edge device will return accepted if the matching occurs; otherwise, it returns rejected. If the matching occurs, the edge device obtains the corresponding ciphertext and forwards it to the corresponding end device.
    3. **Data Decryption**: An end device as the receiver once received the data from the edge device, which means the matching occurs. The end device will conduct the decryption algorithm to recover the message from the ciphertext.

4.2 Concrete Construction

In the following part, we introduce the concrete construction of FADS, which is divided into six algorithms, including Setup, SKGen, RKGen, Enc, Matching, and Dec.

- $\mathsf{Setup}(1^\lambda)$: Given a security parameter 1^λ, the system sets $(p, \mathbb{G}, \mathbb{G}_T, g, e)$ to be the bilinear group applied in our construction, where $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$. Choose $g, h \leftarrow \mathbb{G}$ and $\alpha, \beta \leftarrow$

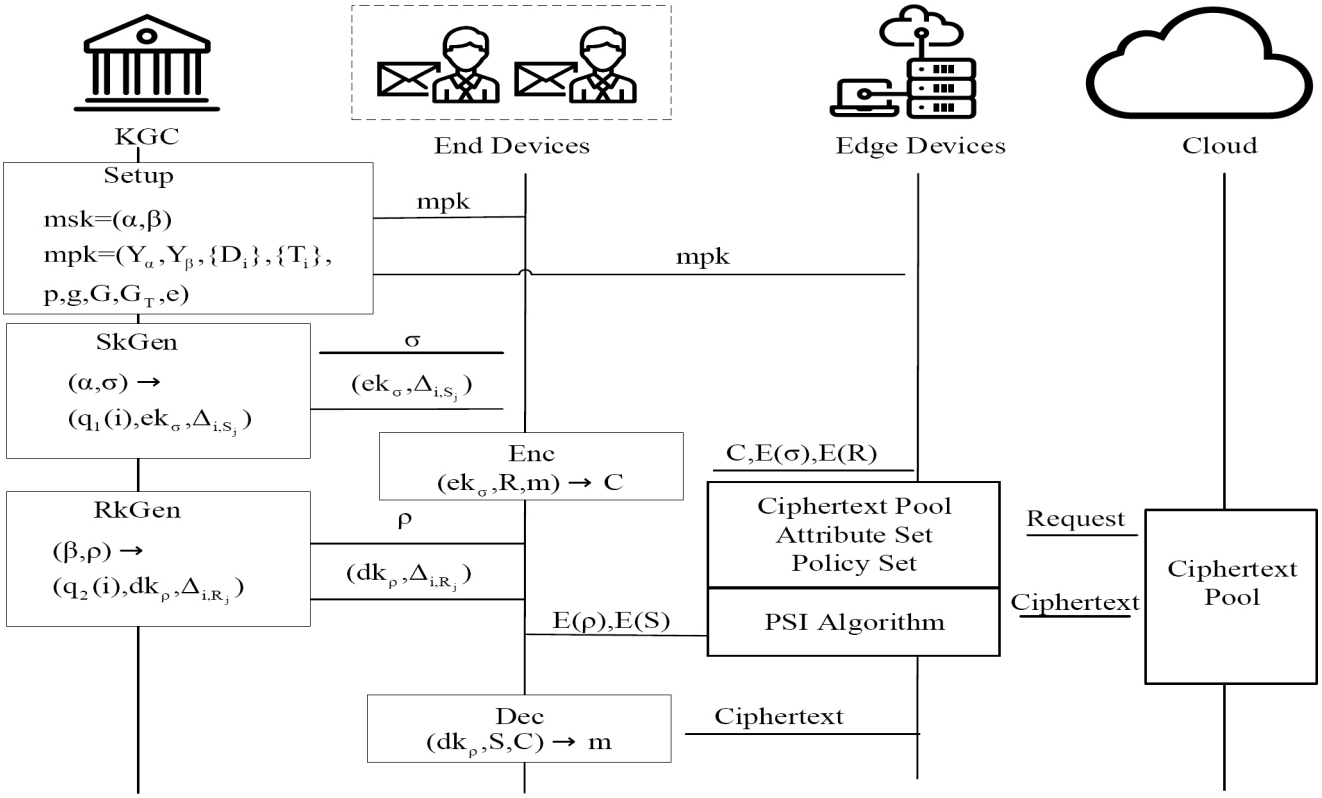

Figure 3. *Workflow of fuzzy matching data sharing scheme.*

$\mathbb{Z}_p$ and set $Y_\alpha = e(g,g)^\alpha, Y_\beta = e(g,g)^\beta$. Then choose n, d and $2n+2$ random values $t_1, \cdots, t_{n+1}, d_1, \cdots, d_{n+1} \leftarrow \mathbb{Z}_p$ and set $T_1 = g^{t_1}, \cdots, T_n = g^{t_{n+1}}, D_1 = g^{d_1}, \cdots, D_n = g^{d_{n+1}}$. The master secret key $\mathtt{msk} = (\alpha, \beta)$. The master public key $\mathtt{mpk} = (Y_\alpha, Y_\beta, \{T_i\}, \{D_i\})$. The public parameters are $(e, \mathbb{G}, \mathbb{G}_T, g, \mathtt{mpk})$.

- $\mathsf{SKGen}(\mathtt{msk}, \sigma)$: The probabilistic algorithm takes the master secret key msk, and the attribute set of sender σ as input. It outputs $\mathtt{ek}_\sigma = \{\mathtt{ek}_i\}$, $\mathtt{ek}_i = g^{\frac{q_1(i)}{t_i}}$, where $i \in \sigma$, and a randomly chosen d-1 degree polynomial q_1 with $q_1(0) = \alpha$. Also, it computes a set of Lagrange coefficient $\{\Delta_{i,\mathbb{S}_j}\}, \mathbb{S}_j \subseteq \sigma$ where a Lagrange coefficient might satisfy the receiver specified policy.
- $\mathsf{RKGen}(\mathtt{msk}, \rho)$: The probabilistic algorithm takes the master secret key msk, and the attribute set of receiver ρ as input. It outputs $\mathtt{dk}_\rho = \{\mathtt{dk}_i\}$,$\mathtt{dk}_i = g^{\frac{q_2(i)}{d_i}}$, where $i \in \rho$, and a randomly chosen d-1 degree polynomial q_2 with $q_2(0) = \beta$. Also, it computes a set of Lagrange coefficient $\{\Delta_{i,\mathbb{R}_j}\}, \mathbb{R}_j \subseteq \rho$ where there might exist a Lagrange coefficient satisfying the sender specified policy.
- $\mathsf{Enc}(\mathtt{ek}_\sigma, \mathbb{R}, m)$: The probabilistic algorithm takes an encryption key $\{\mathtt{ek}_i\}$, a policy of target receiver $\mathbb{R}$ and a message $m \in \{0,1\}^n$ as input. It computes the ciphertext as follows:
  1. Choose random numbers $u, t, r_1, r_2 \in \mathbb{Z}_p$.
  2. Compute $T = g^t, U = g^u, R_1 = g^{r_1}$, and $R_2 = g^{r_2}$.
  3. Compute $P_i = D_i^{r_1}$, where $i \in \mathbb{R}$. And compute $E_i = T_i^{r_2}$.
  4. Compute $W = \prod_{i \in \mathbb{S}_j} ek_i^{\Delta_{i,\mathbb{S}_j}(0)}$, where $|\mathbb{S}_j| \geq d$.
  5. Compute $K_1 = e(R_1, T) \cdot Y_\beta^{r_1}$ and $K_2 = e(R_2, U) \cdot Y_\alpha^{r_2}$.
  6. Compute $V = m \oplus H(e(R_1, T)) \oplus H(e(R_2, U))$.
  7. Output ciphertext

$$C = (T, U, \{P_i\}, \{E_i\}, W, K_1, K_2, V).$$

- $\mathsf{Match}(E(\rho), E(\mathbb{S}), E(\sigma), E(\mathbb{R}))$: Edge devices take E($\rho$), E($\mathbb{S}$), E($\sigma$), and E($\mathbb{R}$) as input. It outputs the current matching result as follows:
  1. Choose $\vec{I} = (I_1, I_2, \cdots, I_n)$ and compute $E(\vec{I})$ as the initial vector in PSI introduced before.
  2. Edge device on sender side computes $E(\mathbb{R} \cap \vec{I})$ and $E(\sigma \cap \vec{I})$. Then, it sends the results to other edge devices.
  3. Edge device on receiver side computes $E(\rho \cap \mathbb{R} \cap \vec{I})$ and $E(\sigma \cap \mathbb{S} \cap \vec{I})$.
  4. If $Dist(\mathbb{S}, \sigma) \geq d \wedge Dist(\mathbb{R}, \rho) \geq d$, return accepted. Otherwise, return rejected.

  Then, only if the matching result is accepted, the edge devices get the corresponding data and return it to the receiver.
- $\mathsf{Dec}(\mathtt{dk}_\rho, \mathbb{S}, C)$: The deterministic algorithm takes the decryption key $\mathtt{dk}_\rho$, a policy of target sender $\mathbb{S}$ and a ciphertext C as input, it decrypts the received ciphertext C as follows:
  1. Parse C as $(T, U, \{P_i\}, \{E_i\}, W, K_1, K_2, V)$.
  2. Compute
  $$g_{T,1} = K_1 / \prod_{i,\mathbb{R}_j} e(P_i, dk_i)^{\Delta_{i,\mathbb{R}_j}(0)},$$
  with $\mathbb{R}_j \subseteq \mathbb{R}, |\mathbb{R}_j| \geq d$.
  3. Compute $g_{T,2} = K_2 / e(\prod_{i,\mathbb{S}} E_i, W)$.
  4. Compute $m = V \oplus H(g_{T,1}) \oplus H(g_{T,2})$.

4.3 Correctness

In this section, we give a detailed proof for the correctness of FADS.

Theorem 1. *If the matching succeeds, where the attributes of the receiver meet the access policy of the sender and the attributes of the sender meet the access policy of the receiver as well, the receiver can correctly recover the message from the ciphertext.*

Proof. The correctness of our construction is oblivious, which depends on the computation of K_1 and K_2. We review K_1, K_2 in Enc as the following:

$$K_1 = e(R_1, T) \cdot Y_\beta^{r_1}, \tag{1}$$

$$K_2 = e(R_2, U) \cdot Y_\alpha^{r_2}. \tag{2}$$

Equation (1) can be transmitted into:

$$\begin{aligned} K_1 = &\ e(R_1, T) \cdot Y_\beta^{r_1} \\ = &\ e(R_1, T) \cdot e(g, g)^{\beta, r_1} \end{aligned}$$

Equation (2) can be transmitted into:

$$\begin{aligned} K_2 = &\ e(R_2, U) \cdot Y_\alpha^{r_2} \\ = &\ e(R_2, U) \cdot e(g, g)^{\alpha, r_2} \end{aligned}$$

In Dec, $g_{T,1}, g_{T,2}$ are as the following equations:

$$g_{T,1} = K_1 / \prod_{i,\mathbb{R}_j} e(P_i, dk_i)^{\Delta_{i,\mathbb{R}_j}(0)}, \tag{3}$$

$$g_{T,2} = K_2 / e(\prod_{i,\mathbb{S}'} E_i, W). \tag{4}$$

Equation (3) can be transmitted into:

$$\begin{aligned} K_1' = &\ g_{T,1} \cdot \textstyle\prod_{i,\mathbb{R}_j} e(P_i, dk_i)^{\Delta_{i,\mathbb{R}_j}(0)} \\ = &\ g_{T,1} \cdot \textstyle\prod_{i,\mathbb{R}_j} e(P_i, dk_i^{\Delta_{i,\mathbb{R}_j}(0)}) \\ = &\ g_{T,1} \cdot \textstyle\prod_{i,\mathbb{R}_j} e(D_i^{r_1}, dk_i^{\Delta_{i,\mathbb{R}_j}(0)}) \\ = &\ g_{T,1} \cdot \textstyle\prod_{i,\mathbb{R}_j} e(g^{d_i \cdot r_1}, dk_i^{\Delta_{i,\mathbb{R}_j}(0)}) \\ = &\ g_{T,1} \cdot \textstyle\prod_{i,\mathbb{R}_j} e(g^{d_i \cdot r_1}, g^{\frac{q_2(i)}{d_i}\Delta_{i,\mathbb{R}_j}(0)}) \\ = &\ g_{T,1} \cdot e(g, g)^{\beta, r_1}. \end{aligned}$$

Equation (4) can be transmitted into:

$$\begin{aligned} K_2' = &\ g_{T,2} \cdot e(\textstyle\prod_{i,\mathbb{S}_j} E_i, W) \\ = &\ g_{T,2} \cdot \textstyle\prod_{i,\mathbb{S}_j} e(T_i^{r_2}, ek_i^{\Delta_{i,\mathbb{S}_j}(0)}) \\ = &\ g_{T,2} \cdot \textstyle\prod_{i,\mathbb{S}_j} e(g^{t_i \cdot r_2}, ek_i^{\Delta_{i,\mathbb{S}_j}(0)}) \\ = &\ g_{T,2} \cdot \textstyle\prod_{i,\mathbb{S}_j} e(g^{t_i \cdot r_2}, g^{\frac{q_1(i)}{t_i}\Delta_{i,\mathbb{S}_j}(0)}) \\ = &\ g_{T,2} \cdot e(g, g)^{\alpha, r_2}. \end{aligned}$$

Suppose that $K_1 = K_1'$ and $K_2 = K_2'$, we can obtain the fact that $e(R_1, T) = g_{T,1}$ and $e(R_2, U) = g_{T,2}$. Then, we can successfully recover the message m from the ciphertext V by calculating

$$m = V \oplus H(g_{T,1}) \oplus H(g_{T,2}).$$

Thus, Theorem 1 is proven.

V. SECURITY ANALYSIS

According to our construction, we have an intuition of the security that the message is semantically secure from the view of a receiver who cannot decrypt the ciphertext. Moreover, since $H(\cdot)$ is close to random distribution, the core idea of the decryption algorithm is to compute symmetric keys $g_{T,1}$ and $g_{T,2}$, motivated by the Fujisaki-Okamoto transformation [45].

Theorem 2. *Let $\mathbb{G}, \mathbb{G}_T$ be two groups of prime order q, and let $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear map. If the decisional MBDH problem is hard in $(\mathbb{G}, \mathbb{G}_T, e)$, and the privacy-preserving set intersection is secure, then our construction is of IND-CCA security against PPT adversary in the random oracle model.*

Suppose that a PPT adversary can break our construction in a random oracle model with advantage ϵ. We can build a simulator $\mathcal{B}$ that can play a game with such an adversary by simulating the system to break the MBDH problem with advantage $poly(\epsilon)$. Let $\mathcal{B}$ preform a sequence of games [46] to prove Theorem 2 as follows.

Proof. Suppose that $\mathcal{B}$ holds an MBDH tuple $(A, B, C, Z) = (g^a, g^b, g^c, e(g,g)^{\frac{ab}{c}})$, $\mathcal{B}$ can construct the simulation, which is indistinguishable from the original scheme from $\mathcal{A}$'s view.

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

Lemma 1. *The advantage of the adversary winning the game in Game 0 is the same as that in the original system. Let $\mathcal{A}$ be an adversary that breaks Game 0. Then, we can take advantage of it $\mathcal{A}$ to break the original scheme.*

$$\mathrm{adv}_{\mathcal{A}}[Game\ 0] = \mathrm{adv}_{\mathcal{A}}[Original]$$

Game 1. Game 1 is different from Game 0 on Setup. $\mathcal{B}$ set the master public key as $Y_\alpha = e(A, g) = e(g, g)^a$, where a is unknown to $\mathcal{B}$, and $Y_\beta = e(A, g)^\beta = e(g, g)^{a\beta}$. For $\forall i \in \mathbb{S}$, it chooses random $t_i \in \mathbb{Z}_p$ and sets $T_i = C^{t_i} = g^{c \cdot t_i}$. For $\forall i \notin \mathbb{S}$, it chooses random w_i and sets $T_i = g^{w_i}$. For $\forall i \in \mathbb{R}$, it chooses random $d_i \in \mathbb{Z}_p$ and sets $D_i = C^{d_i} = g^{c \cdot d_i}$. For $\forall i \notin \mathbb{S}$, it chooses random v_i and sets $D_i = g^{v_i}$. Then, $\mathcal{B}$ gives the public parameters (i.e., master public key, bilinear groups) to $\mathcal{A}$. Here, if the public parameters are indistinguishable to that in Game 0 from the view of $\mathcal{A}$. If the adversary can tell the difference, it will terminate the game and return fail.

Lemma 2. *By difference lemma, if $\mathcal{A}$ can tell the difference between Game 1 and Game 0, we can construct an algorithm that solves discrete logarithm problem with advantage ϵ_{DL}.*

$$|\mathrm{adv}_{\mathcal{A}}[Game\ 1] - \mathrm{adv}_{\mathcal{A}}[Game\ 0]| \leq \epsilon_{\mathrm{DL}}$$

Game 2. Game 2 is different from Game 1 on SKGen and RKGen. $\mathcal{A}$ makes requests for the sender's private keys. Suppose that $\mathcal{A}$ requests a sender's encryption key on the attribute set Θ. We first define three cases in three different set Γ, Γ', Θ: Γ' will be any d-dimension set such that $\Gamma \subseteq \Gamma' \subseteq \Theta$.

- $i \in \Gamma$: $ek_i = g^{s_i}$, where $s_i \in \mathbb{Z}_p$.
- $i \in \Gamma' - \Gamma$: $ek_i = g^{\frac{\lambda_i}{w_i}}$, where $\lambda_i \in \mathbb{Z}_p$.
- $i \notin \Gamma'$:

$$ek_i = (\prod_{j,\Gamma} C^{\frac{t_j s_j \Delta_{j,\Theta}(i)}{w_i}})(\prod_{j,\Gamma'-\Gamma} g^{\frac{\lambda_j \Delta_{j,\Theta}(i)}{w_i}}) Y_\alpha^{\frac{\Delta_{0,\Theta}(i)}{w_i}}.$$

Using the d-1 variables in Γ' and Y to calculate $ek_i, i \notin \Gamma'$ by interpolation on $q_1(x)$ with $q_1(0) = a$. Therefore, $\mathcal{B}$ can construct the sender's encryption key ek_i for sender's attribute set Θ.

$\mathcal{B}$ can simulate the receiver's decryption key using the same method. The decryption key for the receiver with attribute set Θ is as follows:

- $i \in \Gamma$: $dk_i = g^{x_i}$, where $x_i \in \mathbb{Z}_p$.
- $i \in \Gamma' - \Gamma$: $dk_i = g^{\frac{\gamma_i}{v_i}}$, where $\gamma_i \in \mathbb{Z}_p$.
- $i \notin \Gamma'$:

$$dk_i = (\prod_{j,\Gamma} C^{\frac{d_j x_j \Delta_{j,\Theta}(i)}{v_i}})(\prod_{j,\Gamma'-\Gamma} g^{\frac{\gamma_j \Delta_{j,\Theta}(i)}{v_i}}) Y_\beta^{\frac{\Delta_{0,\Theta}(i)}{v_i}}.$$

Here, the sender's encryption key and receiver's decryption key are linearly independent and indistinguishable to that in Game 1 from the view of $\mathcal{A}$. If $\mathcal{A}$ can tell the difference, it will terminate the game and return failed.

Lemma 3. *By difference lemma, if $\mathcal{A}$ can tell the difference between Game 2 and Game 1, we can construct an algorithm that solves discrete logarithm problems with advantage ϵ_{DL}.*

$$|\mathrm{adv}_{\mathcal{A}}[Game\ 2] - \mathrm{adv}_{\mathcal{A}}[Game\ 1]| \leq \epsilon_{\mathrm{DL}}$$

Game 3. Game 3 is different from Game 2 on hash oracle. $\mathcal{A}$ will request the hash oracle to obtain the hash value. For i-th query, $\mathcal{B}$ simulates the hash function by set the hash value as $H(i) = str_i$, where $|str_i| = |m|$. From the view of $\mathcal{A}$, the hash value distribution of the hash oracle is informatively independent of the distribution of the real hash function. If $\mathcal{A}$ can tell the difference, it will terminate the game and return failed.

Lemma 4. *By difference lemma, if $\mathcal{A}$ can tell the difference between Game 3 and Game 2, we can construct an algorithm that finds the collision in the hash oracle with advantage ϵ_{H}.*

$$|\mathrm{adv}_{\mathcal{A}}[Game\ 3] - \mathrm{adv}_{\mathcal{A}}[Game\ 2]| \leq \epsilon_{\mathrm{H}}$$

Game 4. Game 4 is different from Game 3 on Challenge. $\mathcal{A}$ chooses two message m_0 and m_1 to $\mathcal{B}$, where $|m_0| = |m_1|$. $\mathcal{B}$ tosses a coin $b \in \{0, 1\}$, and returns the ciphertext on m_b. $\mathcal{B}$ chooses a d-dimension attribute set of sender $\Theta^*_{\mathrm{se}} \subseteq \Gamma_1 \cap \Gamma_2 \cdots \cap \Gamma_l$, and a d-dimension attribute set of receiver $\Theta^*_{\mathrm{re}} \subseteq \Gamma_1 \cap \Gamma_2 \cdots \cap \Gamma_l$. The ciphertext is as follows:

$$\begin{aligned}
T &= C^t, \\
U &= C^u, \\
P_i &= B^{\beta x_i}, \\
E_i &= B^{s_i \Delta_{i,\Theta^*_{\mathrm{se}}}(i)}, \\
W &= \textstyle\prod_{j,\Theta^*_{\mathrm{se}}} g^{y s_i}, \\
K_1 &= g^{r_1} Z^{\beta}, \\
K_2 &= g^{r_2} Z, \\
V &= m_b \oplus H(e(g,B)^t) \oplus H(e(g,B)^u).
\end{aligned}$$

If $Z = e(g,g)^{\frac{ab}{c}}$ and $r'_1 = \frac{b}{c}$, we have $K_1 = e(g^{r'_1}, C^t) \cdot Z^{\beta} = e(g^{r'_1}, C)^t \cdot e(g,g)^{\beta a \cdot \frac{b}{c}} = (e(g,C)^t Y_{\beta})^{\frac{b}{c}}$, and $P_i = B^{\beta x_i} = g^{b\beta x_i} = g^{c \cdot r'_1 \beta x_i}$. Also, if $r'_2 = y\frac{b}{c}$, we have $K_2 = e(g^{r'_2}, C^u) Z = e(g,C)^{u r'_2} e(g,g)^{a \cdot y \frac{b}{c}} = (e(g,C)^u Y_{\alpha})^{y \frac{b}{c}}$, and $E_i = B^{s_i} = g^{b y s_i} = g^{c \cdot y \frac{b}{c} s_i} = C^{y \frac{b}{c} s_i}$.

If $Z = e(g,g)^z$, since z is randomly chosen from $\mathbb{Z}_p$, $\{P_i\}, \{E_i\}, K_1, K_2$ will be the random elements of the bilinear group from the view of $\mathcal{A}$.

If $\mathcal{A}$ can tell the difference between Game 4 and Game 3, it will terminate the game. Otherwise, $\mathcal{B}$ can solve the given MBDH problem.

If $Z = e(g,g)^z$, the probability of $\mathcal{A}$'s guess $b' = b$ is as follows:

$$\Pr[b' = b | Z = e(g,g)^z] = \frac{1}{2},$$

$$\Pr[b' \neq b | Z = e(g,g)^z] = \frac{1}{2}.$$

If $Z = e(g,g)^{\frac{ab}{c}}$, there is an advantage of $\mathcal{A}$ breaking the game defined as ϵ'. The probability of $\mathcal{A}$'s guess $b' = b$ is as follows:

$$\Pr[b' = b | Z = e(g,g)^{\frac{ab}{c}}] = \frac{1}{2} + \epsilon'.$$

The overall advantage of $\mathcal{B}$ in the MBDH game is

$$\begin{aligned}
\mathrm{Adv}^{\mathrm{MBDH}}_{\mathcal{B}}[b' = b] = \ & |\tfrac{1}{2} \Pr[b' = b | Z = e(g,g)^z] \\
& + \tfrac{1}{2} \Pr[b' = b | Z = e(g,g)^{\frac{ab}{c}}] - \tfrac{1}{2}| \\
= \ & \tfrac{1}{2}(\tfrac{1}{2} + \epsilon' + \tfrac{1}{2}) - \tfrac{1}{2} \\
= \ & \tfrac{1}{2}\epsilon'.
\end{aligned}$$

Lemma 5. *By difference lemma, if $\mathcal{A}$ can tell the difference between Game 4 and Game 3, we can construct an algorithm to solve the MBDH problem with the advantage $\frac{1}{2}\epsilon'$.*

Game 5: $\mathcal{A}$ and $\mathcal{B}$ play the game as before, except that the request attribute set $|\Theta_{\mathrm{re}} \cup \Theta^*_{\mathrm{re}}| \leq d - 1$ and $|\Theta_{\mathrm{se}} \cup \Theta^*_{\mathrm{se}}| \leq d - 1$.

Lemma 6. *By difference lemma, if $\mathcal{A}$ can tell the difference between Game 5 and Game 4, the advantages of $\mathcal{B}$ breaking the DL problem are as Lemma 3 and Lemma 4.*

Combining the proof from Lemma 2 to Lemma 6,

we obtain

$$\mathrm{Adv}_{\mathcal{A}}[b'=b] \leq 4\epsilon_{\mathrm{DL}} + \epsilon_{\mathrm{H}} + \frac{\epsilon'}{2}.$$

Theorem 3. *The security of our proposed FADS implies privacy and authenticity.*

Proof. First, we show how FADS security implies privacy.

Lemma 7. *Suppose that there is an adversary who can break the privacy of our construction. We can take advantage of such an adversary to construct a PPT algorithm for solving the MBDH problem.*

Proof. The proof is similar to that for proving Theorem 2, except that the simulator will give the challenged encryption key to the adversary. The adversary and simulator will do the same things during Game 0, Game 1, Game 2, Game 3, and Game 5. The only difference is of Game 4.

Game 4. The adversary provides two instance I_0, I_1 as $I_0 = (m_0, \mathbb{R}_0, \sigma_0)$ and $I_1 = (m_1, \mathbb{R}_1, \sigma_1)$. The simulator randomly chooses $b \in \{0,1\}$ and computes C^*.

- **Attribute Privacy.** The attribute privacy is based on the security of PSI and the indistinguishability of ciphertext under I_0 or I_1. Take the attribute privacy from sender as an example. Given ek_{σ_b} to the adversary, the adversary can guess $b' = b$ with no advantage over random guess since the $q_1(i)$ is a randomly chosen polynomial, while $q_1(0) = \alpha$ in the real system. In the simulated system, since $\sigma_0, \sigma_1 \subseteq \cup_{i=1}^{q_S} \Gamma_i$, we know that $\forall i \in \sigma_b, ek_i = g^{s_i}$. It is a linearly independent distribution from the view of the adversary. Thus, FADS holds the attribute privacy that

$$\mathrm{Adv}_{\mathcal{A}}^{\mathrm{Att-Pri}}[b'=b|\mathcal{A}^{\mathcal{O}_1,\mathcal{O}_2}(I_0, I_1, C^*)] \leq \epsilon$$

  where ϵ is negligible.

- **Policy Privacy.** In FADS, policy privacy leakage may occur in two phases, i.e., Dec and Match. Firstly, in $\mathsf{Enc}(\mathtt{ek}_\sigma, \mathbb{R}, m)$, the sender specifies policy $\mathbb{R}$ for the receiver and encrypts it as P_i. In $\mathsf{Dec}(\mathtt{dk}_\rho, \mathbb{S}, C)$, the receiver specifies policy $\mathbb{S}$ for the sender, encrypts it as $\prod_{i,\mathbb{S}} E_i$, and attempts to recover m. If the recovery fails, it will not reveal which of $\mathbb{R}$ or $\mathbb{S}$ does not match. If it succeeds, it will not recover $\mathbb{R}$ because the adversary only knows the intersection set between $\mathbb{R}$ and $\mathbb{S}$. Then, since the policy privacy is based on the security of PSI in $\mathsf{Match}(E(\rho), E(\mathbb{S}), E(\sigma), E(\mathbb{R}))$, the edge devices cannot break the policy privacy to get $\mathbb{R}$ and $\mathbb{S}$. Thus, FADS holds the policy privacy that

$$\mathrm{Adv}_{\mathcal{A}}^{\mathrm{Pol-Pri}}[b'=b|\mathcal{A}^{\mathcal{O}_1,\mathcal{O}_2}(I_0, I_1, C^*] \leq \epsilon$$

  where ϵ is negligible.

Overall, the privacy of our construction can be reduced to the security of PSI and the security of our proposed scheme. Here, we can prove that the IND-CCA security of our construction implies privacy.

Lemma 8. *Suppose that there is an adversary that breaks the authenticity of our construction. We can take advantage of such an adversary to construct a PPT algorithm for solving the MBDH problem.*

Proof. The proof is similar to that for proving Theorem 2, except that the adversary finally outputs a forgery C^* computed under the sender, not queried in key generation.

The adversary sends $(C^*, \sigma^*, \mathbb{R})$. The simulator receives such C^*, if the decryption successes, which means $m \neq \perp$, the simulator can parse C^* into $(T, U, \{P_i\}, \{E_i\}, W, K_1, K_2, V)$. The simulator sets $Y_\alpha = Z, Y_\beta = Z^\beta, T_i = A^{w_i}, D_i = B^{v_i}$. Let $q_1(0) = c, q_2(0) = yc$. Here, we just list the components used for reduction. Also, on the other occasion, we will set T_i, D_i as we describe in Theorem 2. It is a simple way to set $q_1(0), q_2(0)$, by using the d-1 points. The detail of the methodology is in Theorem 2. If $Z = e(g,g)^z$, then the adversary will terminate the game. Otherwise, it will continue to play the game with the simulator.

The simulator gets the following equations from the adversary's successful forgery:

$$e(\prod_{i,\mathbb{R}} P_i, dk_i)^{\Delta_{i,\mathbb{R}}(0)} = Z^{\beta r_1}, \tag{5}$$

and

$$e(\prod_{i,\mathbb{S}} E_i, W)^{\Delta_{i,\mathbb{S}}(0)} = Z^{r_2}. \tag{6}$$

We can determine from Equation (5) that (A, B, C, Z) is a MBDH tuple. It implies the existence of Equation (6). Thus, the authenticity of our construction holds if the MBDH problem is hard. Specifically, FADS holds the authenticity that

$$\Pr_{\mathcal{A}}^{\text{Auth}}[m = \text{Dec}(C^*)|\mathcal{A}(\sigma^*, C^*, \mathbb{R})] \leq \epsilon$$

where ϵ is negligible, $m \neq \perp$, and the encryption key on σ^* is never queried during the authenticity game. Therefore, the security of our construction implies authenticity.

Combining Lemma 7 and Lemma 8, we can conclude that the security of FADS implies privacy and authenticity.

VI. PERFORMANCE EVALUATION

In this section, we evaluate the performance of our data sharing scheme with relevant works AFNV19 [8]and CFDS20 [15] by a sequence of experiments. AFNV19 and CFDS20 are typical matchmaking encryption and ME-based data sharing scheme in fog computing, allowing bilateral access control. Therefore, we choose AFNV19 and CFDS20 to conduct the performance comparison.

6.1 Environment Configuration

Our scheme is implemented in Java using JPBC [47]. We choose Type A curve, as symmetric pairings (Type-I), with 80-bit security. The execution environment is conducted with a laptop, which is of an 8th generation Intel Core i7-8550U @ 1.80GHz with 16 GB of RAM.

We executed the experiments 20 times to obtain the average time for each algorithm. We make each end device in the system contain at most 30 attributes ($n \leq 30$), including *ID, name, gender, age, occupation, faculty, organization, suburban, city, country, etc.* To realize fuzzy matching, we set the threshold value d for the number of attributes to be half the size of the policies. Additionally, we leverage the number of attributes to represent the size of policies in our experiments. We evaluate the encryption performance, decryption performance, storage overhead, and communication overhead.

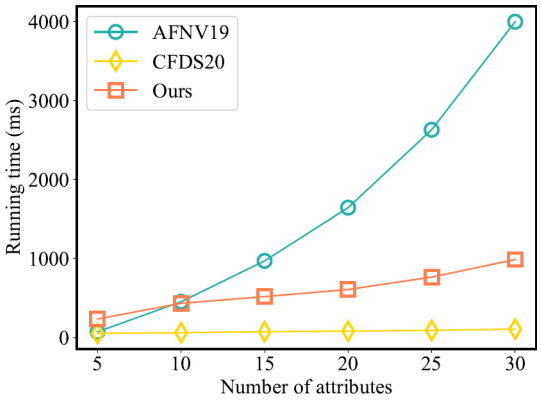

Figure 4. *Encryption performance in FADS.*

6.2 Performance Evaluation

Encryption Performance. As shown in Figure 4, we compare our scheme with AFNV19 and CFDS20 in terms of the encryption running time and the size of policies. In Figure 4, the horizontal axis represents the number of attributes, and the vertical axis represents the average encryption running time overhead. In the encryption process, our scheme, CFDS20, and AFNV19 all generate the ciphertexts based on the bilinear group. Specifically, in the encryption process, the computational cost of our scheme is $(O(1) \times T_{pairing} + O(|\mathbb{S}|) \times T_{multiplication} + O(|\mathbb{R}| + |\mathbb{S}|) \times T_{exponent})$, where $T_{pairing}$ represents the time to perform a pairing operation, $T_{multiplication}$ represents the time to perform a multiplication operation, $T_{exponent}$ represents the time to perform an exponent operation, $|\mathbb{R}|$ represents the size of the sender's policy $\mathbb{R}$, and $|\mathbb{S}|$ represents the size of the receiver's policy $\mathbb{S}$. Compared with CFDS20, our scheme leads to a relatively higher computation cost to support fuzzy matching over the access policies and user's attributes by precomputing some elements, i.e., W. Thus, the running time of encryption in our scheme is slightly higher than that of CFDS20. Since AFNV19 extends directly from an identity-based setting to an attribute-based setting in the encryption process, which incurs a large computation overhead, the running time of the encryption process in AFNV19 is much higher than that of our scheme.

Decryption Performance. As shown in Figure 5, we compare our scheme with AFNV19 and CFDS20 in

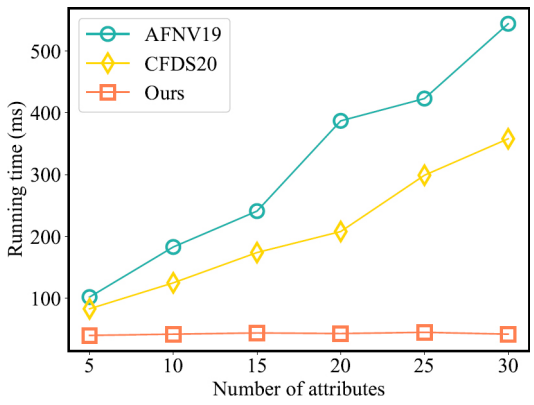

Figure 5. *Decryption performance in FADS.*

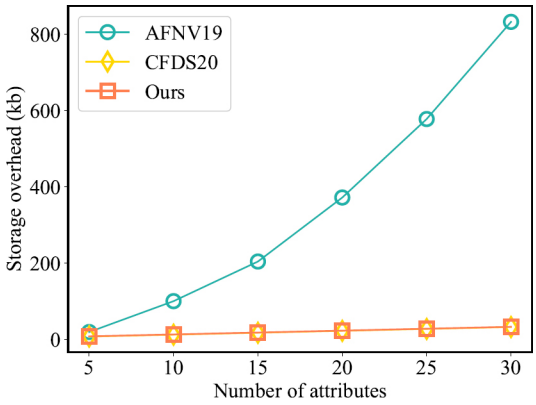

Figure 6. *Storage overhead in FADS.*

terms of the decryption running time and the size of policies. In Figure 5, the horizontal axis represents the number of attributes, and the vertical axis represents the average decryption running time overhead. In AFNV19, the whole process of decryption is executed on the receiver side, which brings a large computation overhead on the receiver side. CFDS20 attempts to reduce the computation overhead on the receiver side by outsourcing the workload of sender verification to edge devices. To further reduce the computation overhead on the receiver side, our scheme outsources the workload of matching between sender and receiver to edge devices. Specifically, in the decryption process, the computational cost of our scheme is $(O(|\mathbb{R}|) \times T_{pairing} + O(|\mathbb{R}| + |\mathbb{S}|) \times T_{multiplication} + O(|\mathbb{R}|) \times T_{exponent})$, where $T_{pairing}$ represents the time to perform a pairing operation, $T_{multiplication}$ represents the time to perform a multiplication operation, $T_{exponent}$ represents the time to perform an exponent operation, $|\mathbb{R}|$ represents the size of the sender's policy $\mathbb{R}$, and $|\mathbb{S}|$ represents the size of the receiver's policy $\mathbb{S}$.

Storage Overhead. As shown in Figure 6, we compare our scheme with AFNV19 and CFDS20 in terms of the storage overhead and the size of policies on the cloud side. In Figure 6, the horizontal axis represents the number of attributes, and the vertical axis represents the average storage overhead. Our scheme has the same performance as the accurate matching type data sharing CFDS20. However, the storage overhead of FADS is much lower than that in AFNV19 because the ciphertext in AFNV19 is the multiply of senders' attributes and receivers' attributes.

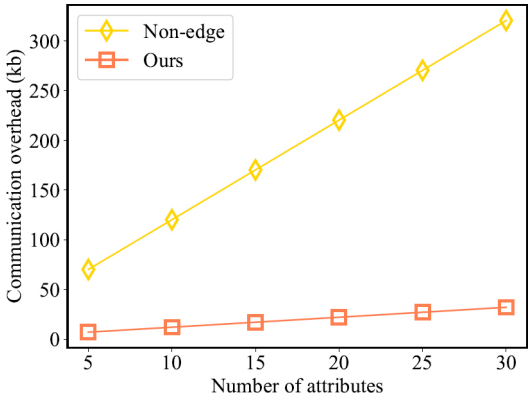

Figure 7. *Communication overhead of FADS.*

Communication Overhead. We evaluate the communication overhead on the receiver side by varying the size of policies in the presence and absence of edge devices. The comparison results are shown in Figure 7, where we set the number of senders as 10 and vary the number of attributes from 5 to 30. Obviously, FADS greatly reduces communication overhead on the receiver side because of the assistance of edge devices.

VII. CONCLUSION

In this paper, we introduce a novel notion of data sharing for the cloud-edge computing environment and provide a concrete construction of FADS with a pairing-based cryptosystem. To the best of our knowl-

edge, FADS is the first data sharing scheme based on fuzzy matchmaking encryption. Our proposed data sharing enables the matching holds with a certain distance of error and allows the policies from both sides to be checked simultaneously without revealing any additional information except the matching holds or not. We give the formal security proof to show the security, privacy, and authenticity. The experiments are conducted to evaluate the performance of our proposed data sharing. By comparing with the existing works, the results indicate that our proposed data sharing is practical.

Our work inspires a few interesting open problems. The first is how to construct a cross-domain fuzzy matching data sharing scheme where the users come from multiple authorities. In real-world applications, it is a common case that users are registered from different authorities. The second problem is to build fuzzy matching data sharing schemes for arbitrary policy to provide fine-grained access control. The third problem is to create more efficient fuzzy matching data sharing schemes with standard assumptions. Furthermore, we should consider including the addressing key escrow [48], key management infrastructure, and revocation [49] efficiently. In addition, applying FADS into other application domains, such as truth discovery [50], task recommendation [51], federated learning [52] is also an interesting and important research direction.

ACKNOWLEDGEMENT

This work is supported by the China Postdoctoral Science Foundation (Grant Nos. 2021TQ0042, 2021M700435, 2021TQ0041), the National Natural Science Foundation of China (Grant No. 62102027), and the Shandong Provincial Key Research and Development Program (2021CXGC010106).

References

[1] B. Flavio, A. M. Rodolfo, *et al.*, "Fog computing and its role in the internet of things," in *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, MCC@SIGCOMM 2012, Helsinki, Finland, August 17, 2012*, 2012, pp. 13–16.

[2] W. Wu, N. Chen, *et al.*, "Dynamic ran slicing for service-oriented vehicular networks via constrained learning," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 7, 2020, pp. 2076–2089.

[3] F. Lyu, J. Ren, *et al.*, "Lead: Large-scale edge cache deployment based on spatio-temporal wifi traffic statistics," *IEEE Transactions on Mobile Computing*, vol. 20, no. 8, 2020, pp. 2607–2623.

[4] Y. Deng, F. Lyu, *et al.*, "Auction: Automated and quality-aware client selection framework for efficient federated learning," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 8, 2021, pp. 1996–2009.

[5] (2020) Edge Computing Market Worth $61.14 Billion By 2028 — CAGR: 38.4%. [Online]. Available: https://www.grandviewresearch.com/press-release/global-edge-computing-market

[6] I. Damgård, H. Haagh, *et al.*, "Access control encryption: enforcing information flow with cryptography," in *Theory of Cryptography - 14th International Conference, TCC 2016-B, Beijing, China, October 31 - November 3, 2016, Proceedings, Part II*, vol. 9986. Springer, 2016, pp. 547–576.

[7] S. Kim and D. J. Wu, "Access control encryption for general policies from standard assumptions," in *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I*, vol. 10624. Springer, 2017, pp. 471–501.

[8] A. Giuseppe, F. Danilo, *et al.*, "Match me if you can: Matchmaking encryption and its applications," in *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part II*, vol. 11693, no. 99, 2019, pp. 701–731.

[9] Z. Chkirbene, A. Mohamed, *et al.*, "Smart edge healthcare data sharing system," in *2020 International Wireless Communications and Mobile Computing (IWCMC)*. IEEE, 2020, pp. 577–582.

[10] Y. Jin, M. Tomoishi, *et al.*, "Secure remote monitoring and cipher data sharing for iot healthcare system with privacy preservation," in *2021 5th International Conference on Cloud and Big Data Computing (ICCBDC)*, 2021, pp. 46–51.

[11] S. Adi, "How to share a secret," *Communications of the ACM*, vol. 22, 1979, pp. 612–613.

[12] P. Jingwen, C. Jie, *et al.*, "Secure data sharing scheme for vanets based on edge computing," *EURASIP Journal on Wireless Communications and Networking*, vol. 2019, 2019, p. 169.

[13] L. Lei, F. Jie, *et al.*, "Blockchain-enabled secure data sharing scheme in mobile-edge computing: An asynchronous advantage actor-critic learning approach," *IEEE Internet Things Journal*, vol. 8, no. 4, 2021, pp. 2342–2353.

[14] Y. Wenti, G. Zhitao, *et al.*, "Secure data access control with fair accountability in smart grid data sharing: an edge blockchain approach," *IEEE Internet Things Journal*, vol. 8, no. 10, 2021, pp. 8632–8643.

[15] X. Shengmin, N. Jianting, *et al.*, "Match in my way: fine-grained bilateral access control for secure cloud-fog computing," *IEEE Transactions on Dependable and Secure Computing*, vol. PP, no. 99, 2020, pp. 1–1.

[16] C. Biwen, X. Tao, *et al.*, "Cl-me: Efficient certificateless

matchmaking encryption for internet of things," *IEEE Internet of Things Journal*, vol. PP, no. 99, 2021, pp. 1–1.

[17] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, vol. 3494. Springer, 2005, pp. 457–473.

[18] Y. Harbi, Z. Aliouat, *et al.*, "Secure data transmission scheme based on elliptic curve cryptography for internet of things," in *2018 International Symposium on Modelling and Implementation of Complex Systems, MISC, Laghouat, Algeria, Dec. 16-18, 2018*, vol. 2, 2019, pp. 229–238.

[19] K. Jiawen, Y. Rong, *et al.*, "Blockchain for secure and efficient data sharing in vehicular edge computing and networks," *IEEE Internet of Things Journal*, vol. 6, no. 3, 2019, pp. 4660–4670.

[20] L. Liu, J. Feng, *et al.*, "Blockchain-enabled secure data sharing scheme in mobile-edge computing: an asynchronous advantage actor–critic learning approach," *IEEE Internet of Things Journal*, vol. 8, no. 4, 2020, pp. 2342–2353.

[21] W. Yang, Z. Guan, *et al.*, "Secure data access control with fair accountability in smart grid data sharing: an edge blockchain approach," *IEEE Internet of Things Journal*, vol. 8, no. 10, 2020, pp. 8632–8643.

[22] D. Francati, A. Guidi, *et al.*, "Identity-based matchmaking encryption without random oracles," in *International Conference on Cryptology in India*. Springer, 2021, pp. 415–435.

[23] A. N. Tentu, P. Paul, *et al.*, "Computationally perfect secret sharing scheme based on error-correcting codes," in *International Conference on Security in Computer Networks and Distributed Systems*. Springer, 2014, pp. 251–262.

[24] R. Cramer, I. B. Damgård, *et al.*, "Linear secret sharing schemes from error correcting codes and universal hash functions," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2015, pp. 313–336.

[25] Z. Jiale, C. Bing, *et al.*, "Data security and privacy-preserving in edge computing paradigm: survey and open issues," *IEEE Access*, vol. 6, 2018, pp. 18 209–18 237.

[26] V. Pande, C. Marlecha, *et al.*, "A review-fog computing and its role in the internet of things," *International Journal of Engineering Research and Applications*, vol. 6, no. 10, 2016, pp. 2248–96 227.

[27] V. Blesson, W. Nan, *et al.*, "Challenges and opportunities in edge computing," in *2016 IEEE International Conference on Smart Cloud, SmartCloud, New York, NY, USA, Nov. 18-20, 2016*, 2016, pp. 20–26.

[28] H. Hongwen, Z. Chengcheng, *et al.*, "A novel secure data transmission scheme in industrial internet of things," *China Communications*, vol. 17, no. 1, 2020, pp. 73–88.

[29] X. Aidong, T. Jie, *et al.*, "Physical layer secure data transmission for mobile edge computing: beamforming and artificial noise," *Journal of Physics: Conference Series*, vol. 1659, no. 1, 2020, p. 012016.

[30] G. Shrivastava, "Secure file transmission scheme based on hybrid encryption technique," *International Journal of Managment It & Engineering*, vol. 2, 2012, pp. 229–238.

[31] K. Lea and S. Dawn Xiaodong, "Privacy-preserving set operations," in *Advances in Cryptology - CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005, Proceedings*, vol. 3621. Springer, 2005, pp. 241–257.

[32] A. Giuseppe, K. Jonathan, *et al.*, "Secret handshakes with dynamic and fuzzy matching," in *Proceedings of the Network and Distributed System Security Symposium, NDSS 2007, San Diego, California, USA, 28th February - 2nd March 2007*, vol. 7, no. 24, 2007, pp. 43–54.

[33] G. R. Blakley, "Safeguarding cryptographic keys," in *Managing Requirements Knowledge, International Workshop on*. IEEE, 1979, pp. 313–313.

[34] R. Cramer, I. Damgård, *et al.*, "General secure multi-party computation from any linear secret-sharing scheme," in *Advances in Cryptology - EUROCRYPT 2000, International Conference on the Theory and Application of Cryptographic Techniques, Bruges, Belgium, May 14-18, 2000, Proceeding*, vol. 1807. Springer, 2000, pp. 316–334.

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

[36] 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.

[37] A. Beimel, O. Farràs, *et al.*, "Linear secret-sharing schemes for forbidden graph access structures," in *Theory of Cryptography - 15th International Conference, TCC 2017, Baltimore, MD, USA, November 12-15, 2017, Proceedings, Part II*, vol. 10678. IEEE, 2017, pp. 394–423.

[38] M. Ben-Or, S. Goldwasser, *et al.*, "Completeness theorems for non-cryptographic fault-tolerant distributed computation," in *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*. ACM, 1988.

[39] K. Okada and K. Kurosawa, "Mds secret-sharing scheme secure against cheaters," *IEEE Transactions on Information Theory*, vol. 46, no. 3, 2000, pp. 1078–1081.

[40] F. Song, Z. Qin, *et al.*, "Privacy-preserving task matching with threshold similarity search via vehicular crowdsourcing," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 7, 2021, pp. 7161–7175.

[41] C.-N. Yang, X. Wu, *et al.*, "Intragroup and intergroup secret image sharing based on homomorphic lagrange interpolation," *Journal of Information Security and Applications*, vol. 61, no. 11, 2021, p. 102910.

[42] F. Miao, Y. Yu, *et al.*, "Grouped secret sharing schemes based on lagrange interpolation polynomials and chinese remainder theorem," *Security and Communication Networks*, vol. 2021, no. 99, 2021, p. 99.

[43] C. Y. Andrew, "Protocols for secure computations (extended abstract)," in *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982*, vol. 99, 1982, pp. 160–164.

[44] W. Wenli, L. Shundong, *et al.*, "Privacy-preserving mixed set operations," *Information Sciences*, vol. 525, 2020, pp. 67–81.

[45] F. Eiichiro and O. Tatsuaki, "Secure integration of asymmetric and symmetric encryption schemes," in *Advances in*

Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings, vol. 1666. IACR, 1999, pp. 537–554.

[46] S. Victor, "Sequences of games: A tool for taming complexity in security proofs," *IACR Cryptology ePrint Archive*, vol. 2004, 2004, p. 332.

[47] D. Angelo and I. Vincenzo, "jpbc: Java pairing based cryptography," in *Proceedings of the 16th IEEE Symposium on Computers and Communications, ISCC 2011*, 2011, pp. 850–855.

[48] G. Sanjam, H. Mohammad, *et al.*, "Registration-based encryption from standard assumptions," in *Public-Key Cryptography - PKC 2019 - 22nd IACR International Conference on Practice and Theory of Public-Key Cryptography, Beijing, China, April 14-17, 2019, Proceedings, Part II*, vol. 11443, 2019, pp. 63–93.

[49] X. Zhiqian and M. M. Keith, "Dynamic user revocation and key refreshing for attribute-based encryption in cloud storage," in *11th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom 2012, Liverpool, United Kingdom, June 25-27, 2012*, 2012, pp. 844–849.

[50] Z. Chuan, Z. Liehuang, *et al.*, "Reliable and privacy-preserving truth discovery for mobile crowdsensing systems," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 3, 2019, pp. 1245–1260.

[51] Z. Chuan, Z. Liehuang, *et al.*, "Location privacy-preserving task recommendation with geometric range query in mobile crowdsensing," *IEEE Transactions on Mobile Computing*, 2021.

[52] Z. Weiting, Y. Dong, *et al.*, "Optimizing federated learning in distributed industrial IoT: A multi-agent approach," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, 2021, pp. 3688–3703.

Biographies

Chuan Zhang received his Ph.D. degree in computer science from Beijing Institute of Technology, Beijing, China, in 2021. He is currently an assistant professor at School of Cyberspace Science and Technology, Beijing Institute of Technology. His research interests include secure data services in cloud computing, applied cryptography, machine learning, and blockchain.

Mingyang Zhao received his B.S. degree in Beijing Institute of Technology in 2021. He is currently working towards the master degree in School of Cyberspace Science and Technology, Beijing Institute of Technology. His research interests include applied cryptography, cloud security, and blockchain.

Yuhua Xu is currently an undergraduate student in School of Computer Science and Technology, Beijing Institute of Technology. She is currently working at the research laboratory of advanced network and data security in School of Cyberspace Science and Technology, Beijing Institute of Technology. Her research interests include applied cryptography and blockchain.

Tong Wu received her Ph.D. degree in computer science from University of Wollongong, Australia, in 2020. She is currently a postdoctoral research fellow at School of Cyberspace Science and Technology, Beijing Institute of Technology. Her research interests include applied cryptography, cloud security, and blockchain security.

Yanwei Li received his Ph.D degree in electronic science and technology from Tsinghua University, China, in 2013. He is currently a senior engineer at National Computer Network Emergency Response Technical Team/Coordination Center of China. His current research interests include secure data services in cloud computing and big data security.

Liehuang Zhu (Senior Member, IEEE) received his Ph.D. degree in computer science from Beijing Institute of Technology, Beijing, China, in 2004. He is currently a professor at the School of Cyberspace Science and Technology, Beijing Institute of Technology. His research interests include security protocol analysis and design, group key exchange protocols, wireless sensor networks, and cloud computing.

Haotian Wang is currently an undergraduate student in the College of Arts and Science, at University of Pennsylvania, U.S. His research interests include blockchain-based applications, political economy, and biochemistry.