IEEE Access on Blockchain,Issue Date:09-Mar-2021

Blockchain-Assisted Collaborative Service Recommendation Scheme With Data Sharing

BIWEI YAN[1], ANMING DONG[2], BAOBAO CHAI[2], YUBING HAN[2], (Member, IEEE), GUANGLIN ZHOU[2], AND FANGXIN ZHAO[2]

[1]School of Mathematical Sciences, Qufu Normal University, Qufu 273165, China
[2]School of Computer Science and Technology, Qilu University of Technology (Shandong Academy of Sciences), Jinan 250353, China

Corresponding author: Anming Dong (anmingdong@qlu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 61832012, Grant 61672321, Grant 61771289, and Grant 61701269.

ABSTRACT With the rapid development of cloud computing, a large number of web services have been emerging quickly, which brings a heavy burden for users to choose the services they preferred. In order to suggest web services for users, recommendation algorithms are needed and many of them have been investigated recently. However, most of the existing recommendation schemes are based on centralized historical data, which may lead to single point of failure. Generally, the data contains a lot of sensitive information that cloud may expose the privacy of users, which makes most cloud platforms reluctant to share their own data. In order to solve the above issues, the secure data sharing among cloud platforms is necessary for better recommendation, which can maximize the profits. In this paper, we propose a blockchain-assisted collaborative service recommendation scheme (*BC-SRDS*). Specifically, we adopt the ciphertext-policy attribute-based encryption (CP-ABE) algorithm to encrypt the data, which ensures the data confidentiality and realizes secure data sharing. Then, we utilize the blockchain to share data, such that the DoS attack, DDoS attack and single point of failure can be avoided. Meanwhile, the data integrity, tampering-proof of data are guaranteed through the blockchain. And we use locality-sensitive hashing algorithm to recommend the services for users. Finally, it is proved through the security analysis that *BC-SRDS* is capable of achieving data confidentiality, data integrity and tampering-proof. A series of experiments show that *BC-SRDS* achieves better recommendation accuracy compared with the existing schemes.

INDEX TERMS Collaborative service recommendation, blockchain, data sharing, ciphertext-policy attribute-based encryption.

I. INTRODUCTION

For the rapid development of the Internet and computer technology, a large number of network information services have entered people's daily life, providing users with many conveniences. Meanwhile, due to the rapid growth of Internet users, the information generated by users also presents an explosive growth, leading to the problem of ''information overload''. It is an important challenge for producers and consumers in today's society: for Internet service providers, it is very difficult for Internet service providers to select the services through user information quickly and make their services popular with the public. For Internet users, it is not easy to select the services they are interested in from the vast amount of information, which requires a lot of time and energy.

In order to solve above problems, a lot of recommendation algorithms have been proposed, among which collaborative filtering recommendation is a common method. It provides personalized recommendation for target user according to the scoring records of resources. Moreover, the algorithm has a good recommendation effect and is widely used in personalized sites, e-commerce, and other fields. Although the collaborative filtering recommendation algorithm has high recommendation accuracy, it still faces a series of challenges: the data used in the collaborative filtering recommendation algorithm is often stored on the centralized server, so there is no historical data for new users or new items to refer to, and it may encounter cold start problem and cannot complete the recommendation. However, by collecting the data of users on

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

different platforms, the problem of cold start can be solved effectively. For example, user *A* has called the services of Amazon and user *B* has called the services of IBM. If *A* and *B* are similar users, they can recommend services to *A* by analyzing the services of *B*, or recommend services to *B* by analyzing the services of *A*. Nonetheless, a malicious recommendation may lead to data abuse or privacy leakage issues. Therefore, in order to protect users' privacy information, Amazon and IBM are reluctant to share user data with each other. In this case, we cannot obtain similar users of new users, which greatly reduces the quality of recommendation. Besides, assume that two platforms agree to share data, and obtain the data from different platforms to improve the recommendation accuracy. But there still exists the problem that the data stored in different distributed platforms greatly increases the communication overhead between cloud platforms, which can not response quickly when online.

In order to solve the above problems, in this paper, we propose a novel blockchain-assisted collaborative service recommendation scheme with data sharing (*BC-SRDS*), with which every platform can share their data securely based on blockchain. And locality-sensitive hash (LSH) is an efficient data search algorithm that can quickly find similar data, therefore, the LSH is adopted to perform quick recommendation.

The main contribution can be summarized as follows:

1) To the best of our knowledge, many recommendation algorithms are based on centralized data, but little of them adopt blockchain to realize the recommendation. Moreover, it is difficult to combine distributed cloud platforms to share their data with each other for recommendation because of users' privacy. In order to realize better recommendation, in this paper, we introduce blockchain to provide a secure sharing environment for accurate recommendation.
2) Different from most existing recommendation works, we adopt the CP-ABE (ciphertext-policy attribute-based encryption) to promote the data sharing among the cloud platforms and combine blockchain to ensure the security of data provenance, avoid the risk on the failure of single point, improve data integrity, and defend against DoS or DDoS attacks.
3) Based on the real distributed QoS (quality of service) dataset WS-DREAM,[1] the experimental results show the effectiveness of *BC-SRDS*. The analysis of the metrics such as CPU consumption, memory consumption, throughput, latency, MAE (Mean Absolute Error), RMSE (Root Mean Square Error), the usage of gas and the number of similar neighbors proves that *BC-SRDS* can significantly improve the accuracy and gain more profits.

The rest of this paper is organized as follows. Section II elaborates the related work in this paper. And in section III, we present the preliminaries of this paper. Then, section IV introduces the overview of our proposed scheme. Section V proposes a novel blockchain-assisted collaborative service recommendation in detail. Section VI analyzes the security and evaluates the performance of the proposed scheme. Finally, section VII summarizes the paper.

[1] https://wsdream.github.io/

II. RELATED WORK

The recommendation algorithms are mainly divided into three types: content-based recommendation algorithm [1], [2], collaborative filtering recommendation algorithm [3], and hybrid recommendation algorithm [4]. Among them, the collaborative filtering algorithm, also known as cooperative filtering algorithm or social filtering algorithm, is one of the most successful recommendation algorithms and is widely used in the field of government, logistics, and education. The early collaborative filtering recommendation algorithm is based on the user, which analyzes the historical data of services that the target user has visited and calculates the similarity by using a certain similarity measurement method to obtain similar users of the target user. Finally, the algorithm recommends the services that visited by similar users but not visited by the target users to the target users, so as to realize personalized recommendation [5]. However, with the rapid growth of network scale and Internet users, there are many problems in the user-based collaborative filtering recommendation algorithm, such as scalability, cold start, and sparsity problems. Therefore, Sarwar *et al.* [6] proposed a collaborative filtering algorithm based on items, which mainly calculates the similarity between items by analyzing the user's historical data. Because the number of users are much larger than the number of items, the computing of similarity between items is less sparse than that between users. Meanwhile, because the properties of items are fixed, we can calculate the similarity between items offline. Moreover, the calculation performance between items is high. Compared with collaborative filtering algorithm based on users, the recommendation quality is greatly improved. However, the item-based collaborative filtering algorithm also brings new challenges: Firstly, the algorithm does not consider the difference between users, thus, the recommendation quality is poor; secondly, when a new item is added to the system, it is difficult to recommend the items that are similar to the new item to users, because it is not scored or there are few scores about the new item. In order to solve these problems, Jiang *et al.* [4] proposed a hybrid recommendation algorithm, which takes the advantages of the two algorithms that are item-based collaborative filtering algorithm and user-based collaborative filtering algorithm. They use the characteristics of attributes of users or items to obtain a better recommendation effect. However, the algorithm is based on specific applications and it is difficult to transplant and expand. Cheng *et al.* [7] proposed a method to improve the recommendation quality by introducing metric learning into collaborative filtering algorithm. By measuring the distance between target user and candidate set, it separates the candidate set. And the items that have high similarity to user preference are close to the users while the items that have

low similarity to user preference are far away from users, which reduces the influence of sparse data on the recommendation. Although the algorithm can alleviate the influence of sparsity on the recommendation, but it does not propose an effective method to deal with sparse data. In [8], Mnih proposed the matrix decomposition algorithm, which is one of the most popular collaborative filtering algorithms based on model. Compared with user-based collaborative filtering algorithm, it shows better performance in dealing with sparse problems. In [9], Yu *et al.* added the dimension of geographic location. They pointed out that users in similar geographical location usually invoked the same service, and they often have the same service requirements. Besides, the quality of web service is highly related to time. Thus, a time-aware QoS prediction scheme is proposed in [10]. According to the geographical distribution, the QoS information with a timestamp is collected to predict the quality of service, and a better prediction result is obtained. Considering that the quality of recommendation is related to the personal preference of users, and the time and geographical location of the recommendation cannot meet the needs of recommendation. Therefore, Li *et al.* integrated the sorting learning technology into the recommendation algorithm to recommend the items that match the user's requirements by building the user preference demand model in [11].

In order to form a more effective web service recommendation, users need to provide more information. However, the data of these users is managed by the service provider, and some sensitive information involving users' privacy may be at risk of being leaked. On the one hand, centralized servers may leak users' privacy due to the wrong operation. On the other hand, if the centralized server is invaded by malicious nodes, user information may be sold to third parties, leading to data abuse. Therefore, in [12], Badsha *et al.* developed a privacy protection protocol, through which the missing QoS value is predicted. And it is based on user location and QoS data to realize web service recommendation. Meanwhile, the protocol encrypts location information and QoS data, which can recommend suitable services to users as well as realize the protection of users' privacy. In [13], Zhu *et al.* adopted data obfuscation technology to hide the real service quality data by adding obfuscated data to the original data, achieving privacy protection. However, the introduction of confusing data in this scheme can lead to a decrease in the accuracy of recommendation. Moreover, the data obfuscation also brings additional time cost. In addition, technologies such as anonymization [14], randomization [15], and cryptography [16] have also been proposed for various privacy protection collaborative filtering algorithm to protect users' privacy [17]. In addition, Zheng *et al.* proposed a privacy-preserved a framework in IIoTs for data sharing in [18]. In [19], Wang *et al.* used blockchain to share information, but they do not consider the security of shared data. Moreover, a blockchain-aided searchable attribute-based encryption for cloud-IoT was proposed by Liu *et al.* to realize the data sharing [20]. And in [21], Cai *et al.* proposed a private and efficient mechanism for data uploading.

III. PRELIMINARIES

In this section, we will give a brief review on preliminaries of our scheme.

A. BILINEAR MAPS

Let $\mathbb{G}_0$, $\mathbb{G}_1$ be two multiplicative cyclic groups of the same prime order p; g_1 is a generator of $\mathbb{G}_0$ and g_2 is an element of $\mathbb{G}_0$, respectively. A bilinear map e: $\mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$ has the following three properties:

1) *Bilinearity*: $\forall$a,b $\in \mathbb{Z}_p$ and $g_1, g_2 \in \mathbb{G}_0$, we have $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$.
2) *Non-degeneracy*: $\forall g_1, g_2 \in \mathbb{G}_0$, $e(g_1, g_2) \neq 1$, which means that the map does not send any pair in $\mathbb{G}_0 \times \mathbb{G}_0$ to the identity in $\mathbb{G}_1$.
3) *Computability*: There is an efficient algorithm to compute $e(g_1, g_2)$ for all $g_1, g_2 \in \mathbb{G}_1$.

B. CIPHERTEXT-POLICY ATTRIBUTE-BASED ENCRYPTION (CP-ABE)

In this section, we adopt the original CP-ABE scheme [22] for data sharing. The CP-ABE scheme consists of four algorithms: **Setup**, **Encrypt**, **KeyGen**, and **Decrypt**.

Setup$(1^\lambda) \rightarrow (PK, MSK)$. The setup algorithm takes the security parameter λ as the input. It outputs a master key MSK and the public parameters PK.

Encrypt$(PK, m, \mathbb{A}) \rightarrow CT$. The encryption algorithm takes a message m, the public parameters PK, and an access structure $\mathbb{A}$ over the universe of attributes as input. The algorithm will encrypt m and generate a ciphertext CT, which can be decrypted by the user whose attributes set satisfies the access structure. Assume that $\mathbb{A}$ is implicitly contained in the ciphertext.

KenGen$(MSK, S) \rightarrow SK$. The key generation algorithm takes a set of attributes S and the master key MSK as input and outputs a private key SK.

Decrypt$(PK, CT, SK) \rightarrow m$. The decryption algorithm takes the public parameters PK, a ciphertext CT which contains an access structure $\mathbb{A}$, and a secret key SK as input, where SK is a secret key for a set $\mathbb{A}$ of attributes. If the set S with attributes satisfies the access structure $\mathbb{A}$, the algorithm will decrypt the ciphertext and return a message m.

C. BLOCKCHAIN TECHOLOGY

Blockchain was originally proposed by Nakamoto in 2008 [23], which is a distributed ledger and can solve distributed problems effectively. At present, blockchain is mainly classified into three categories: the public blockchain, the private blockchain, and the consortium blockchain. For the public blockchain, its unique feature is completely decentralized so that any node can join the blockchain to access the data on the blockchain. To the opposite, the private blockchain is

produced by an institution or an organization, only nodes belonging to the institution or the organization can join the private blockchain and access. Finally, when it comes to the consortium blockchain, it is usually used for more than two institutions or organizations. If a node wants to join the blockchain network, it must have been authenticated before.

Different from the public blockchain and the private blockchain, the consortium blockchain integrates the advantages of both. Therefore, it preserves the privacy of the private blockchain as well as maintains decentralization of the public blockchain. Only authenticated nodes can successfully join the consortium blockchain. Combining the advantages of fast transaction processing, flexible transactions, multi-centralization, and high scalability, the application scope is wider. Therefore, the consortium blockchain is chosen to construct our scheme. Besides, smart contract is a core technology on blockchain which can perform trusted transaction automatically without the third parties. In particular, it always performs operations according to pre-defined rules. Therefore, anyone cannot violate the smart contract. And smart contract exists on the blockchain in digital form, so it cannot be tampered with due to the characteristics of blockchain. Moreover, it is a participant in the system that can response the request, store and receive value as well as send the data and value. In our scheme, we adopt the two blockchain platforms, Hyperledger Fabric[2] and Ethereum,[3] to simulate the proposed scheme. We mainly use Hyperledger Fabric to deploy our scheme, but Hyperledger Fabric has no function of token. Therefore, we use the Ethereum to measure the cost of uploading the data for cloud platform.

D. LOCALITY-SENSITIVE HASHING (LSH)

In traditional hashing algorithms, even similar data may be very different after being hashed. Namely, only the original data is the same, the hash result in traditional hash algorithm will be the same. Otherwise, the results of traditional hash algorithm will be very different even if there are little changes in the original data. However, different from the traditional hashing algorithms, in locality-sensitive hashing algorithm, similar data may be still similar after locality-sensitive hashing with high probability. Therefore, we choose the locality-sensitive hashing to find the similar items. Locality-Sensitive hashing (LSH) was proposed by Gionis in 1999 [24]. As a hashing algorithm designed for processing the similarity search problem of high-dimensional data, the natures of LSH are that: 1) LSH guarantees that similar data in the original space is still similar in the new space after locality-sensitive hashing with high probability. 2) Data which is not similar in the original space will be similar in the new space after locality-sensitive hashing with low probability.

[2]https://www.hyperledger.org/projects/fabric

[3]https://ethereum.org/en/

IV. SCHEME OVERVIEW

A. OVERVIEW

In this section, we propose a blockchain-assisted collaborative service recommendation scheme with data sharing. Different from the scheme[4] in [25], we adopt CP-ABE to realize secure data sharing so that we can share data securely while guaranteeing that the cloud platforms can control their own data. CP-ABE embeds the access strategy into the ciphertext, which means that the data owners can determine that who can access these user data. Further, it prevents user historical service data from being illegally accessed. Therefore, the scheme can meet the needs for security of user historical service data. Moreover, our scheme also introduces blockchain to realize data sharing. Its features of tampering-proof, decentralization can further ensure the security of ciphertexts. Our scheme can easily detect whether the data has been tampered with. Therefore, our scheme is secure for recommendation. Moreover, the system model of our scheme is shown in FIGURE 1. In this scheme, it mainly includes three layers: User layer, Data sharing layer, and Data layer.

1) User layer: This layer mainly includes various devices belonged to users, through which users can access the web services. They will produce large-scale data on the cloud platforms. Note that these users are distributed in different platforms.
2) Data sharing layer: The cloud platforms that agree to share the data construct a consortium blockchain, on which they can share and utilize message. They will record the shared data on the consortium blockchain. Using the data on consortium blockchain, the platforms can merge it with their own data to perform accurate recommendation, obtaining more profits. Every platform maintains a public ledger together.
3) Data layer: In this layer, assisted with the cloud servers, we store the large files (e.g. videos, images) on it, because blockchain is not suitable for storing large amounts of data.

As shown in FIGURE 1, users firstly send a request to a cloud platform for a number of services they preferred. Along with the request, users produce lots of data, which are stored on the cloud platforms. In order to provide a good quality of experience and gain more benefits, it is a good idea for different cloud platforms to share data collaboratively. In detail, the scheme constructs the consortium blockchain, to which every cloud platform joins. The cloud platforms share their data collaboratively. And they can also obtain the shared data to realize accurate recommendation. Before sharing user service data to the blockchain, they perform CP-ABE encryption algorithm to encrypt these data, which can protect the users' privacy information effectively. Because only the platform that satisfies the attribute requirements can obtain the encrypted data. Next, the encrypted data CT is uploaded

[4]It is our conference-version paper of this work, which has been presented in the International Conference on Wireless Algorithms, Systems, and Applications (WASA).

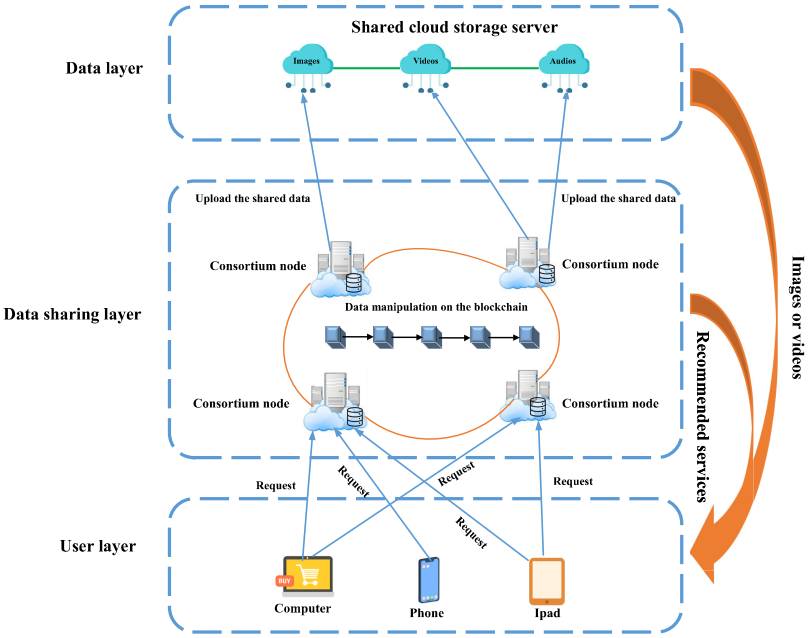


FIGURE 1. The model of service recommendation using blockchain.

to the blockchain in the form of transaction i. Then, the transactions will be broadcast on the consortium blockchain for all the nodes to verify and invoke, and the transaction will be packaged into the block. After successful mining, the transaction will be recorded on the consortium blockchain. However, for the restriction of the blockchain, the large files (e.g. images and videos) are uploaded to the cloud servers. Then, the cloud platforms can obtain the encrypted data CT, if their attributes are in the authorized sets, they can decrypt the shared data. Finally, the cloud platform will merge their data with the decrypted data m to make an accurate recommendation for users.

B. SECURITY DEFINITION

1) SECURITY MODEL

Our proposed scheme satisfies the security requirement of data confidentiality. Through the work [22], we can prove the data confidentiality by the security game. Next, we describe the security model of CP-ABE scheme [22].

1) **Setup.** The algorithm is run by the challenger $\mathcal{C}$, which sends the public parameters and PK to the adversary $\mathcal{A}$.
2) **Phase 1.** The adversary $\mathcal{A}$ makes some repeated queries to obtain the private keys that are corresponding to the attributes set $S_1, \ldots, S_{q_1}$.
3) **Challenge.** $\mathcal{A}$ submits two equal length messages m_0 and m_1. Then, $\mathcal{A}$ queries for a challenge access structure $\mathbb{A}^*$ such that none of the attributes set $S_1, \ldots, S_{q_1}$ satisfy the access structure $\mathbb{A}^*$. $\mathcal{C}$ randomly chooses a bit b, which is used to encrypt the message m_b under $\mathbb{A}^*$. The obtained ciphertext CT^* is sent to $\mathcal{A}$.
4) **Phase 2.** Phase 1 is repeated with the restriction that none of attributes set $S_{q_1+1}, \ldots, S_q$ satisfy the access structure $\mathbb{A}^*$ corresponding to the challenge.
5) **Guess.** $\mathcal{A}$ outputs a guess b' of b.

The probability $Pr[b' = b] - \frac{1}{2}$ is defined as $\mathcal{A}$'s advantage to win this game.

Definition 1: A CP-ABE scheme is secure if all polynomial time adversaries have at most a negligible advantage in the above game.

2) SECURITY PROPERTIES

For our scheme, we assume that all the cloud platforms on the consortium blockchain are "honest-but-curious", they will neither abuse nor disclose shared data. Consequently, every cloud platform can use shared data honestly. Meanwhile, the following security properties are necessary for the proposed scheme.

- **Tamper-proofing.** All nodes should hardly tamper with the shared data. Once the shared data is tampered with by a node, the data can be detected.
- **Avoiding single point of failure.** The scheme should meet the requirement that when a cloud platform was attacked, other cloud platforms can also run normally.
- **Data integrity.** When the shared data is tampered with, the scheme can quickly determine whether the data is complete.
- **Defending against DoS or DDoS attacks.** If an attacker sends huge amount of requests to perform the DoS (Denial of Service) or DDoS (Distributed Denial of Service) attack to the system, the scheme can defend against DoS or DDoS attacks to ensure the normal working of the system.

V. BLOCKCHAIN-ASSISTED COLLABORATIVE SERVICE RECOMMENDATION SCHEME

In this section, according to the scheme overview, we introduce the construction of our scheme in detail. In order to better understand the scheme, we firstly present the scheme in briefly. In our scheme, the CP-ABE [22] is used to encrypt the shared data to ensure the security of data provenance. And the blockchain is used to realize the encrypted data sharing. The shared data will be backed up to each node in the blockchain, which avoids the single point of failure issues and ensures data integrity. Meanwhile, only users who satisfy the access tree can decrypt shared data to ensure data confidentiality, and avoid the security issues caused by key transmission in traditional symmetric key encryption. Data sharing between cloud platforms through blockchain can reduce communication overhead, ensure data quality, and improve the accuracy of service recommendation. The original locality-sensitive algorithm in [26] is adopted to achieve the item matching, which speeds up the item matching speed and improves the users' experience. The symbols are defined in Table 1.

TABLE 1. Notations.

Symbol	Semantics
u	the number of users in each cloud platform
I_j	the j-th service
$I_{i,j}^q$	the j-th quality dimension q of I_j in platform cp_i
l	the number of the hash functions
cp_i	the i-th cloud platform
N_c	the number of cloud platforms
n	the number of web services
L	the number of hash tables
$\mathbb{G}_0$	$\mathbb{G}_0$ is a multiplicative cyclic group
p	p is the prime order of $\mathbb{G}_0$
λ	the security parameter of the system
t	the t-th hash table
g_1	the generator of $\mathbb{G}_0$
g_2	it is an element in $\mathbb{G}_0$
MSK	it is the master secret key
PK	it is the public key
SK	it is the private key
$\mathcal{T}$	a tree access structure
x	x is a node in $\mathcal{T}$
j	an attribute belonged to the user
S	a set of attributes
CT	the ciphertext of plaintext m

A. ALGORITHM DEFINITION

The proposed scheme mainly contains the following 7 algorithms: **Setup**, **KenGen**, **ShareData**, **Decrypt**, **Build indices for each service**, **Get the final service index by integrating indexes offline** and **Service recommendation**.

(**1**) **Setup**. The algorithm is run by the smart contract, which takes λ as input, and outputs the public key and a master key MSK.

(**2**) **KenGen**. This key generation algorithm is executed by the key generation center. It inputs the attributes S of the cloud platform, and outputs a key that can identify with the attributes for each cloud platform.

(**3**) **ShareData**. The cloud platform performs the algorithm, which takes the plaintext m, the public key PK and the access structure $\mathcal{T}$ as input. Meanwhile, the algorithm outputs the ciphertext CT that will be uploaded to consortium blockchain.

(**4**) **Decrypt**. The algorithm is run by the cloud platform and it inputs the public key PK, the ciphertext CT and the private key SK, then it will decrypt CT to obtain the plaintext m.

(**5**) **Build indices for each service**. The algorithm is run by the cloud platform. It takes all data as input, and outputs the indices of web services.

(**6**) **Get the final service index by integrating indexes offline**. The cloud platform takes the web service I's sub-index as input, and outputs the final service index of the web service I.

(**7**) **Service recommendation**. The algorithm takes a target service of the target user as input, and outputs items that is similar to the target service that will be recommended to the target user.

B. SCHEME CONSTRUCTION

In this section, we introduce our scheme including the following 7 algorithms in detail.

(**1**) **Setup**. The setup algorithm mainly generates a public key PK and a master key MSK. The algorithm will firstly choose a bilinear group $\mathbb{G}_0$ with prime order p, and we denote its generator as g. And then it randomly chooses two elements $\alpha, \beta \in \mathbb{Z}_p$. The public key PK is published as follow:

$$\mathrm{PK} = \{\mathbb{G}_0, g, h = g^{\beta}, f = g^{\frac{1}{\beta}}, e(g,g)^{\alpha}\} \tag{1}$$

and we present the master key $MSK = (\beta, g^{\alpha})$. And then, the PK is broadcast to the consortium blockchain in the form of transaction. Besides, the cloud platforms package the transaction into a block and after the consensus algorithm, the transaction will be recorded on the consortium blockchain for public. Each cloud platform can access the public key. While the parameter MSK is kept secret.

(**2**) **KenGen**. The key generation algorithm will generate a unique private key SK for each platform, it takes a set of attributes S as input and outputs a key that identifies with that attributes set. The algorithm chooses a random $r \in \mathbb{Z}_p$ at first, and then selects a random $r_j \in \mathbb{Z}_p$ for each attribute $j \in S$. Finally, it computes the SK as

$$\mathrm{SK} = \left(D = g^{(\alpha+r)/\beta}, \forall j \in S : D_j = g^r \cdot H(j)^{r_j}, D_j' = g^{r_j}\right) \tag{2}$$

Then, the key generation center sends the SK to each cloud platform.

(**3**) **ShareData**. The algorithm encrypts a message m which represents history of services accessed by a user on the platform under the tree access structure $\mathcal{T}$. For each node x (including the leaves) in the tree $\mathcal{T}$, a polynomial q_x is selected by the algorithm. The cloud platform can choose these polynomials in a top-down manner. Firstly, they start

from the root node R in the $\mathcal{T}$ and denote the degree d_x of the polynomial q_x as $d_x = k_x - 1$.

Then, the cloud platform chooses a random $s \in \mathbb{Z}_p$ and they set $q_R(0) = s$ from the root node R. And it also randomly chooses d_R other points of the polynomial q_R to define it completely. Next, it sets $q_x(0) = q_{\text{parent }(x)}(\text{ index }(x))$ and chooses d_x other points randomly to completely define q_x for any other node x. Assuming that Y is the leaf nodes set of $\mathcal{T}$. The cloud platform can construct the ciphertext based on the access structure $\mathcal{T}$ and get the corresponding ciphertext CT as follows:

$$\begin{aligned}\mathrm{CT} = \Big(\mathcal{T}, \tilde{C} = \mathrm{m} \cdot e(g,g)^{\alpha s}, C = h^s, \\ \forall y \in Y: \quad C_y = g^{q_y(0)}, C'_y = H(\mathrm{att}(y))^{q_y(0)}\Big) \end{aligned} \tag{3}$$

Then, the cloud platform uploads CT to other cloud platforms through the consortium blockchain. Then, we combine the Ethereum and obtain the Algorithm 1. As shown in Algorithm 1, the cloud platform uploads the CT to the consortium blockchain. The ciphertext CT is encapsulated as a transaction. Then, the cloud platform packages them into a block and through the consensus algorithm, the transaction will be recorded on the consortium blockchain.

(**4**) **Decrypt**. Our decryption algorithm is described as a recursive algorithm. The cloud platforms can get the shared data that they wanted from the consortium blockchain. The algorithm performs the recursive computing to obtain relevant services of users and achieve accurate service recommendation. In addition, the decryption algorithm is performed off-chain by cloud platforms. For convenient to explain, the algorithm introduces the simplest form of the decryption algorithm.

A recursive algorithm DecryptNode(PK,CT,SK) is defined at first, which needs to input a ciphertext

$$\mathrm{CT} = \left(\mathcal{T}, \tilde{C}, C, \forall y \in Y : C_y, C'_y\right) \tag{4}$$

of user service history data, a private key SK of a specific platform that is associated with a set of attributes S, and a random node x in the access tree $\mathcal{T}$. Only when the node x is indeed a leaf node, we make i = attr(x) and define as follows: On the one hand, if $i \in S$, then

$$\begin{aligned}\mathrm{DecryptNode}(\mathrm{CT}, \mathrm{SK}, x) &= \frac{e\left(D_i, C_x\right)}{e\left(D'_i, C'_x\right)} \\ &= \frac{e\left(g^r \cdot H(i)^{r_i}, h^{q_x(0)}\right)}{e\left(g^{r_i}, H(i)^{q_x(0)}\right)} \\ &= e(g,g)^{rq_x(0)}\end{aligned} \tag{5}$$

On the other hand, for the case that $i \notin S$, we define $\mathrm{DecryptNode}(CT, SK, x) = \perp$. And we take the recursive case when x is a non-leaf node into consideration. The execution process of the $\mathrm{DecryptNode}(CT, SK, x)$ algorithm is as follows: For all children nodes z of a non-leaf node x, it executes $\mathrm{DecryptNode}(CT, SK, z)$ and represents the output as F_z. The S_x denotes an arbitrary k_x-sized set of child nodes z such that $F_z \neq \perp$. The node was unsatisfied and $\perp$ will be returned by the Decrypt algorithm if there is no such set can be satisfied.

Otherwise, we compute F_x by the equation (6) and return the result. Since the DecryptNode algorithm has been defined above, we can continue to define the decryption algorithm. The decryption algorithm starts from the root node R of the access tree $\mathcal{T}$ by invoking the DecryptNode function. If the attribute S satisfies the access tree, we set $A = \mathrm{DecryptNode}(CT, SK, r) = e(g,g)^{rq_R(0)} = e(g,g)^{rs}$.

$$\begin{aligned}F_x &= \prod_{z \in S_x} F_z^{\Delta_{i,S'_x}(0)}, \quad \text{where } \begin{matrix} i = \mathrm{index}(z) \\ S'_x = \{\,\mathrm{index}\,(z) : z \in S_x\}\end{matrix} \\ &= \prod_{z \in S_x} \left(e(g,g)^{r \cdot q_z(0)}\right)^{\Delta_{i,S'_x}(0)} \\ &= \prod_{z \in S_x} \left(e(g,g)^{r \cdot q_{\text{parent }(z)}(\text{ index }(z))}\right)^{\Delta_{i,S'_x}(0)} \text{(by construction)} \\ &= \prod_{z \in S_x} e(g,g)^{r \cdot q_x(i) \cdot \Delta_{i,S'_x}(0)} \\ &= e(g,g)^{r \cdot q_x(0)} \quad \text{(using polynomial interpolation)}\end{aligned} \tag{6}$$

The platform computes

$$\frac{\tilde{C}}{e(C,D)/A} = \frac{\tilde{C}}{e(h^s, g^{(\alpha+r)/\beta})/e(g,g)^{rs}} = m \tag{7}$$

to obtain encrypted user service message m.

After the above decryption operation, the platform can obtain the shared data of a specific user from the consortium blockchain.

(**5**) **Build indices for each service**. The cloud platform will integrate the shared data from the consortium blockchain with their own users' historical service data m and recommend the services for users. Moreover, the vector $\vec{I}$ denotes the web service I on the platform cp_i. The q represents the quality dimension of I. The vector is expressed as $I\vec{(i)} = (I^q_{i,1}, \ldots, I^q_{i,n})$. It should be noted that if $I^q_i = 0$, the service I_i has not been called by the user. Then, the following equation in (8) is used to calculate the hash of $I\vec{(i)}$ with the LSH algorithm. Besides, $\vec{v} = (v_1, \ldots, v_n)$ means an u-dimensional vector, which is selected randomly from [-1,1]; the symbol $\circ$ represents the dot product between two vectors.

$$h(\vec{I(i)}) = \begin{cases} 1 & If \;\; \vec{I(i)} \circ \vec{v} > 0 \\ 0 & If \;\; \vec{I(i)} \circ \vec{v} \leq 0 \end{cases} \tag{8}$$

Replace different vector v and repeat the above steps l times, then we can get the sub-index of the user on the platform cp_i and represent as $H_i(I) = \{h_{i,1}(\vec{I(i)}), \ldots, h_{i,l}(\vec{I(i)})\}$

(**6**) **Get the final service index by integrating indexes offline**. With the web service I's sub-index that we have obtained, we can get the final service index $H(I) = \{H_1(I), \ldots, H_{N_c}(I)\}$. Next, we construct a mapping relationship that is denoted as "$I \rightarrow H(I)$" and then record it in the hash table.

(**7**) **Service recommendation**. Based on the LSH algorithm, if the following equation (9) holds, we can get the most

of web services I_x that is similar to the target service I_q.

$$H(I_x) = H(I_q) \tag{9}$$

However, the LSH algorithm is a probability-based neighbor search technology, so we cannot deduce that I_x is not similar to the target service I_q. Therefore, we construct L hash tables so that the recommendation accuracy can be more accurate. Moreover, when the following condition (10) holds, we can conclude that service I_x and I_q are similar.

$$\exists\, Table_t\ (t \in \{1, \ldots, L\}), \quad satisfies\ H(I_x)_t = H(I_q)_t \tag{10}$$

After that, the similar services I_x that the target user has never invoked are put into the set *sim_web*. Then, if the target user has several target services, in order to find the optimal services, we repeat (8)-(10) and put them into the set *sim_web* (if a service appears serval times in *sim_web*, then we adopt the average service quality to measure the service quality). Then, the optimal service is chosen according to the average service quality from *sim_web*. Finally, by the following equation (11), where the optimal algorithm computes the average service quality, and then the optimal service I_x will be recommended to the target user according to the average service quality.

$$I_x^q = optimal(I_k^q | I_x \in sim\_web) \tag{11}$$

It is noted that we use the consortium blockchain to construct a data sharing platform, on which the cloud platforms share the data collaboratively. Specifically, the constructed consortium blockchain is used not only for recording the public parameters (e.g. public keys), but also storing the shared data of the cloud platforms. With the assistance of the consortium blockchain, the public parameters and the shared data cannot be easily tampered with, such that the concerns about privacy leakage is dispelled and a cloud platform is more willing to share their data with other cloud platforms. As a result, each collaborative cloud platform can benefit from this securely data sharing. For example, the accuracy of the recommendation systems can be improved by the data sharing, since more data can be utilized to achieve high-quality recommendations.

In this paper, it is considered to be "collaborative" if a cloud platform share the data on the consortium blockchain with the others. We note that every cloud platform that satisfied the access tree can obtain the shared data on the consortium blockchain. Since the data are shared by all the cloud platforms, they can be accessed by every collaborative cloud platform. Therefore, this is a kind of collaborative data sharing, and each collaborative cloud platform can benefit from this data sharing.

VI. SECURITY ANALYSIS AND PERFORMANCE EVALUATION

In this section, we first present a security analysis and then give a performance evaluation about *BC-SRDS*.

Algorithm 1 Upload Shared Data to Blockchain

Require: The service data *Data* of each user
Ensure: Gas usage of Users *gasUsage*

```
for j to u do
    c_j = encrypt(Data[j]) //encrypt the service of each user
    uploadedData[j] = b2a_hex(c_j) //convert the ciphertext to 16-ary that is required by the blockchain
    transaction[j] = buildTr(uploadedData[j])
    gasUsage.append(w3.eth.esGas(transaction[j])) // estimate the Gas that the transaction[j] cost
    tx_hash = w3.eth.sendTr(transaction[j]) //send the transaction to blockchain
    receipt = w3.eth.waitForTr(tx_hash, 15) // obtain the receipt from the blockchain
    if receipt then
        continue
    else
        failureTr.append(transaction[j])
        return tr[j] send failure
    end if
end forreturn gasUsage
return s
```

A. SECURITY ANALYSIS

In this section, we present a brief security analysis of our scheme from two aspects: security proof of data confidentiality and security properties analysis.

1) SECURITY PROOF OF DATA CONFIDENTIALITY

Now, we give the security proof of data confidentiality in detail.

Theorem 1: If there is no polynomial time adversary who can break the security of CP-ABE with non-negligible advantage, then our scheme can guarantee the data confidentiality.

Proof 1: We construct an adversary $\mathcal{A}$ that can break the CP-ABE [22] with non-negligible advantage. $\mathcal{A}$ plays the following game with the CP-ABE scheme.

- **Setup.** The adversary $\mathcal{A}$ obtains the public key $PK = \{\mathbb{G}_0, g, h = g^{\beta}, f = g^{\frac{1}{\beta}}, e(g,g)^{\alpha}\}$ of CP-ABE, but the master key $MSK = (\beta, g^{\alpha})$ is secret for the adversary $\mathcal{A}$.
- **Phase 1.** The adversary $\mathcal{A}$ makes a private key query to the challenger $\mathcal{C}$ with a set S_i $(1 \le i \le q_1)$, and then $\mathcal{A}$ gets the private key corresponding to S_i. $\mathcal{C}$ will response $\mathcal{A}$ with the following SK_i:

$$SK_i = \Big(D = g^{(\alpha+r)/\beta}, \quad \forall j \in S : D_j = g^r \cdot H(j)^{r_j}, D'_j = g^{r_j}\Big)$$

Then, for each private key query corresponding to S_i, all the SK_is are returned to $\mathcal{A}$. And all the queried attributes set $\mathcal{R} = \{S_1, S_2, \ldots, S_i\}$ consist of $\mathcal{A}$'s attributes set.

- **Challenge.** After the Phase 1 completed, an access structure $\mathbb{A}^*$ is output. Moreover, $\mathcal{A}$ challenges the two messages $m_0, m_1 \in \mathbb{G}$. For the queried attributes set such as $\mathcal{S}^{(q_1)} = \{S_1, S_2, \ldots, S_i\}$, they do not satisfy $\mathbb{A}^*$. $\mathcal{A}$ sends the two messages $m_0, m_1 \in \mathbb{G}$ to the challenger $\mathcal{C}$, and $\mathcal{C}$ responses it with the ciphertext:

$$\mathrm{CT}^* = \Big(\mathbb{A}^*, \tilde{C} = \mathrm{m} \cdot e(g,g)^{\alpha s}, C = h^s, \\ \forall y \in Y: \quad C_y = g^{q_y(0)}, C'_y = H(\mathrm{att}(y))^{q_y(0)}\Big)$$

  Finally, $\mathcal{C}$ return the ciphertext CT^* to $\mathcal{A}$.
- **Phase 2.** $\mathcal{A}$ performs the queries that are not queried in Phase 1. $\mathcal{A}$'s queries for attributes set $S_{q_1+1}, \ldots, S_q$ should not satisfy $\mathbb{A}^*$. $\mathcal{A}$ performs the same operation as he/she does in Phase 1, and obtains SK_is corresponding to S_i $(q_1 + 1 \le i \le q)$.
- **Guess.** Finally, $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$, and the advantage of the adversary $\mathcal{A}$ against the proposed scheme is:

$$Adv\mathcal{A} = |Pr[b = b'] - \frac{1}{2}|$$

  Above equation denotes $\mathcal{A}$ has non-negligible advantage against the CP-ABE scheme, which ensures the data confidentiality in our scheme. The proof of the theorem is completed.

2) SECURITY PROPERTIES ANALYSIS

In this section, we describe the analysis of security properties in detail.

3) TAMPER-PROOFING

In *BC-SRDS*, if an adversary intends to tamper the shared data on consortium blockchain, the characteristics of blockchain determine that he/she has to control at least 51% nodes in the blockchain. Assume that ϵ means the probability of the adversary to fully control a node and there are m nodes on the consortium blockchain. More than that, we also assume that an adversary has controlled w nodes and the height of the current block is h. In this condition, we construct a random oracle and any adversary can make q queries randomly. If a adversary intends to tamper k blocks, he needs to make $(k-h)(m/2+1)q$ queries. The probability p of successfully tampering with the shared data is $p > 1/(\epsilon^{m/2+1}(k-h)/(m/2+1)q)$. Obviously, we can conclude that when m is large, it is negligible for adversaries to tamper the shared data.

4) AVOIDING SINGLE POINT OF FAILURE

In our proposed scheme, compared with the traditional recommendation schemes, we apply blockchain to our recommendation scheme. All cloud platforms that join to the consortium blockchain are regarded as nodes, the data on these nodes is distributed equally and all of them store a complete copy of the transaction on the consortium blockchain.

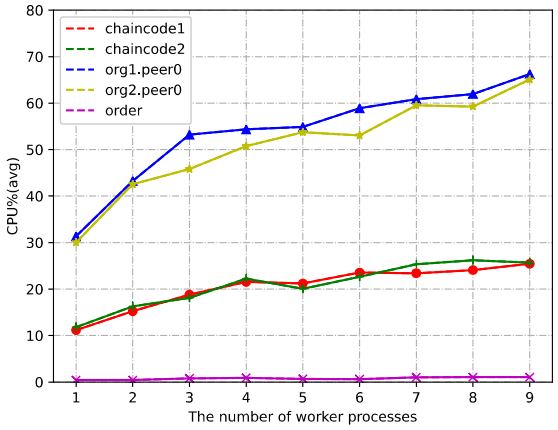


FIGURE 2. The CPU consumption of each container in blockchain.

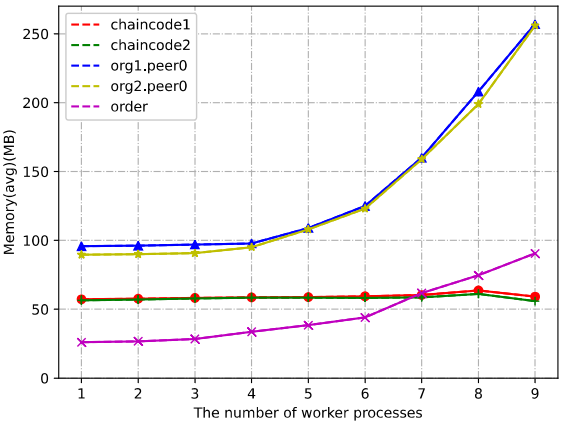


FIGURE 3. The memory consumption of each container in blockchain.

They are dispersed and even if one of the nodes fails, it will not affect the data stored in other nodes. Moreover, the data of user service will not loss or damage. It perfectly solves the single point of failure problem and ensures the continuous working of our recommendation scheme.

5) DATA INTEGRITY

In consortium blockchain, each block contains the hash value of their previous blockchain hash value and the hash value is computed with the transaction, nonce, timestamp by the one-way hash function such as SHA-256. If an adversary tampers with the transaction data on the consortium blockchain, it can be detected. Because the hash value of every subsequent block will be changed. Thus, if an adversary tries to tampered with the transaction, he/she needs powerful computing resources to all hash value of the following blocks so that he/she cannot be detected.

6) DEFENDING AGAINST DoS OR DDoS ATTACKS

Centralized servers are extremely vulnerable to DoS or DDoS attacks due to limited ports. But in the blockchain, all data are distributed and stored on each node in the blockchain. Even if a node is attacked or data is lost, but when it comes back online, they can resynchronize the data through a consensus

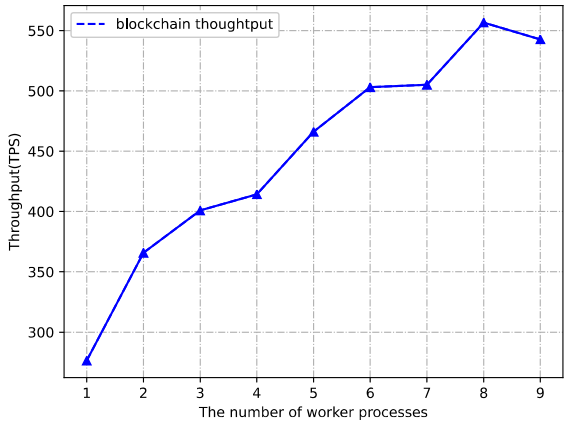


FIGURE 4. The throughput of blockchain.

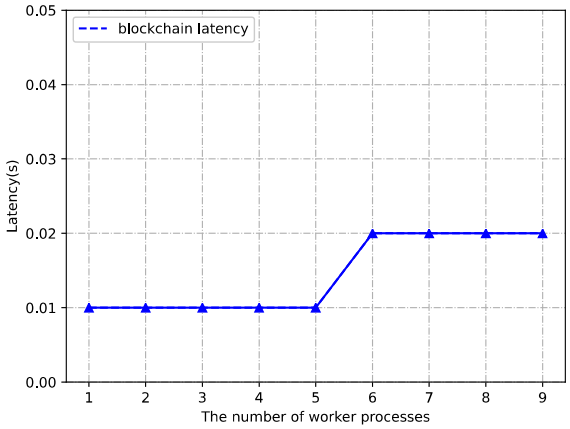


FIGURE 5. The latency of blockchain.

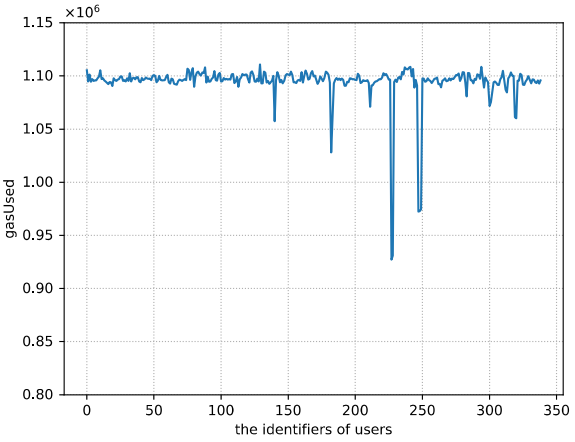


FIGURE 6. The gas usage of each user.

algorithm to ensure data consistency and integrity, which can effectively resist DoS or DDoS attacks.

B. PERFORMANCE EVALUATION

In this section, we deploy a series of experiments with the dataset WS-DREAM, which is a web service QoS dataset for recommendation and was collected in August 2009. And WS-DREAM is a real QoS data from 339 user on 5825 web services (including 1974675 response time and throughput records of service invocations) in different countries

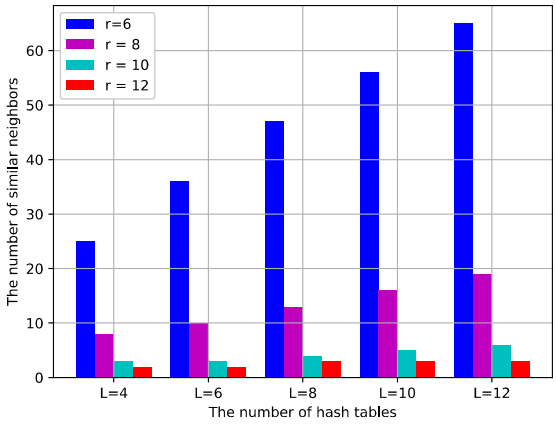


FIGURE 7. The number of matched neighbors.

(i.e. USA and Canada), which constructs a 339-by-5825 user-item matrix based on the response time or throughput. Moreover, PlanetLab[5] nodes in 30 countries are used to simulate the 339 users, while the 5825 real-world web services are crawled from the Internet, which are distributed at 73 countries.

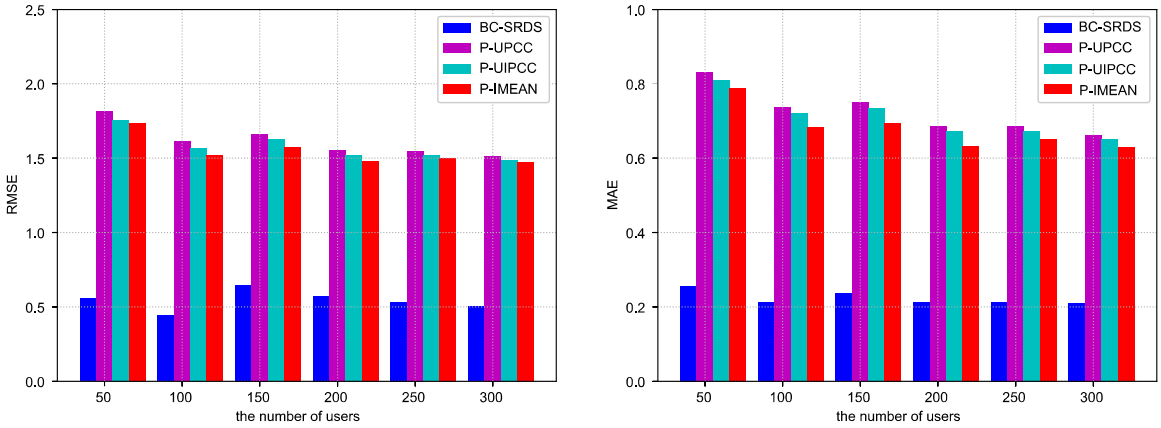
In *BC-SRDS*, we compare our scheme with the other three existing schemes: P-UIPCC, P-UPCC, P-IMEAN in [13]. We adopt the metrics of CPU consumption, memory consumption, throughput, latency, MAE, RMSE, the usage of gas and the number of matched neighbors to evaluate our scheme.

Our experiments are run on Ubuntu 16.04 operating system based on linux virtual machine. Also, python 3.5, web3.py, and Ethereum are used to realize our experiments. We simulate four nodes on Ethereum to denote four cloud platforms. When the cloud platforms share data on Ethereum, it will cost gas. With the gas costs, we can easily help enterprises calculate profits, and facilitate their decision. Moreover, we also deploy the experiments on the Hyperledger Fabric, in which we construct an order node, a client, two organizations, and two peer nodes. Each organization includes a peer node and the two peer nodes are in the same chaincode. Meanwhile, the two peer nodes share a ledger. The client will invoke the Fabric SDK (e.g. Go) to interact with the blockchain. This blockchain construction method with Fabric has the characteristics of scalability, security, and fast transactions, which is very suitable for enterprise-level applications. Moreover, all members must have permission and verifiable identities to join the consortium blockchain which ensures the node credibility.

In order to test the performance of our constructed blockchain, we adopt the caliper, which is a blockchain performance benchmark framework[6] released by the Hyperledger Foundation. It can test different blockchain frameworks such as Hyperledger Sawtooth, Hyperledger Fabric and Ethereum. Next, we show the performance of our blockchain solutions.

[5]PlanetLab(https://www.planet-lab.org) includes 1341 nodes at 654 global sites, which is open platform for research.

[6]https://hyperledger.github.io/caliper/



(a) RMSE (b) MAE

FIGURE 8. The accuracy of service recommendation.

The resource consumption with respect to CPU and memory. The FIGURE 2 and FIGURE 3 show that CPU consumption and memory consumption of each container on blockchain. From FIGURE 2, we can see that as the number of worker processes increases, the consumption of two chaincodes and two peer nodes also increases. Because when the number of worker processes increases, the resource that related to the operation will increase, so the consumption of CPU also increases. However, there is little influence on the order node, because we test the Fabric network under solo mode. It is same as memory consumption shown in FIGURE 3, therefore, we won't explain the FIGURE 3.

The throughput of blockchain. In FIGURE 4, it depicts the throughput of our constructed blockchain. When we increase the number of worker processes, the throughput of our constructed blockchain basically increases. As the number of worker processes increases, the ability of blockchain to process transactions also improves. Therefore, the transaction throughput is promoted.

The latency of blockchain. Through the FIGURE 5, we can see that the average latency of blockchain is very low. When the number of worker processes is varied from 1 to 9, the average latency in processing transactions on our constructed blockchain is varied from 10ms to 20ms. In FIGURE 5, it is obvious that the latency reaches the millisecond level, which can meet performance requirement of our scheme.

Gas usage of BC-SRDS. It shows the gas usage of each user in our scheme as shown in FIGURE 6. When cloud platforms share users' data on the consortium blockchain in form of transactions that contain 5825 services, it consumes the gas on Ethereum. Through the Algorithm 1, we can obtain gas usage of uploading the data for a user. Moreover, in FIGURE 6, the maximum consumption of gas for a user is about 1105352 gas, and the minimum consumption of gas is about 927048 gas. In October 22, 2020, 1 *eth* $\approx$ 412 USD, 1 *gas* $\approx$1 *wei* (0.000000001 *eth*). Therefore, it will cost at most 1105352 *gas* $\approx$ 0.46 USD for sharing users' data. With these gas costs, it is worthy for a cloud platform to share data with other cooperation platforms because they cost less but obtain more profits.

The number of matched neighbors. In FIGURE 7, it is number of matched neighbors with respect to the number of matched neighbors. The main parameters are set as follows: L is varied from 4 to 12, $u = 339$, and $n = 5825$. The experimental results show that when the hash function is fixed, as the number of hash tables increases, the number of matched neighbors will gradually increase. Because when the hash tables increase, the constraints have become looser, so the matched neighbors grow. However, when the hash table is fixed, as the hash function increases, the number of matched neighbors decreases. Because the search condition becomes strict.

Accuracy among four schemes. Accuracy of the four schemes are measured by the metrics RMSE and MAE respectively. The main parameters are set as follows: $L = 4, l = 10$ and u is varied from 50 to 300. The experiment results are shown in FIGURE 8, in which $n = 5825$ holds. As we can see in FIGURE 8, *P-UPCC* has the lowest accuracy in both (a) and (b) (that is, the RMSE and MAE values all are the highest) because the data of service has been confused. The result of *BC-SRDS* is better than other three schemes. That is because only the "most similar" services are returned for service recommendation in *BC-SRDS*; therefore, the accuracy of our scheme is significantly improved. In addition, in FIGURE 8, we can also observe that the recommendation accuracy does not change rapidly with the increase of the number of users, and it basically maintains at a stable level.

VII. CONCLUSION

In this paper, we propose a service recommendation scheme named *BC-SRDS* which not only supports the data sharing among different platforms based on the consortium blockchain but also provides an accurate service recommendation for users. Moreover, to guarantee data security, we encrypt the data by CP-ABE algorithm before sharing

them to other cloud platforms. Through blockchain, the cloud platforms can easily get the shared data and use it to maximize the profits, meanwhile, the DoS attack, DDoS attack and single point of failure are avoided. The security analysis show that *BC-SRDS* is capable of achieving data confidentiality, data integrity and tampering-proof. Finally, we evaluate our scheme based on WS-DREAM and carry out a series of experiments. The experimental results show that *BC-SRDS* can achieve a higher accuracy than other three schemes. Besides, its gas cost is entirely acceptable for the cloud platforms. Moreover, from metrics of the resource consumption, throughput and latency on consortium blockchain, we can conclude that our scheme based on the consortium blockchain is feasible.

REFERENCES

[1] A. Popescul, L. H. Ungar, D. M. Pennock, and S. Lawrence, "Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments," 2013, *arXiv:1301.2303*. [Online]. Available: http://arxiv.org/abs/1301.2303

[2] G. Arora, A. Kumar, G. S. Devre, and A. Ghumare, "Movie recommendation system based on users similarity," *Int. J. Comput. Sci. Mobile Comput.*, vol. 3, no. 4, pp. 765–770, 2014.

[3] J. Bobadilla, F. Ortega, and A. Hernando, "A collaborative filtering similarity measure based on singularities," *Inf. Process. Manage.*, vol. 48, no. 2, pp. 204–217, Mar. 2012.

[4] C. Jiang, R. Duan, H. K. Jain, S. Liu, and K. Liang, "Hybrid collaborative filtering for high-involvement products: A solution to opinion sparsity and dynamics," *Decis. Support Syst.*, vol. 79, pp. 195–208, Nov. 2015.

[5] H. G. Rong, S. X. Huo, C. H. Hu, and J. X. Mo, "User similarity-based collaborative filtering recommendation algorithm," *J. Commun.*, vol. 35, no. 2, pp. 16–24, 2014.

[6] K.-Y. Chung, D. Lee, and K. J. Kim, "Categorization for grouping associative items using data mining in item-based collaborative filtering," *Multimedia Tools Appl.*, vol. 71, no. 2, pp. 889–904, Jul. 2014.

[7] C.-K. Hsieh, L. Yang, Y. Cui, T.-Y. Lin, S. Belongie, and D. Estrin, "Collaborative metric learning," in *Proc. 26th Int. Conf. World Wide Web*, 2017, pp. 193–201.

[8] A. Mnih and R. R. Salakhutdinov, "Probabilistic matrix factorization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2008, pp. 1257–1264.

[9] C. Yu and L. Huang, "A Web service QoS prediction approach based on time- and location-aware collaborative filtering," *Service Oriented Comput. Appl.*, vol. 10, no. 2, pp. 135–149, Jun. 2016.

[10] X. Zhang, Z. Wang, W. Zhang, and F. Yang, "A time-aware QoS prediction approach to Web service recommendation," in *Proc. 4th Int. Conf. Comput. Eng. Netw.* Cham, Switzerland: Springer, 2015, pp. 739–748.

[11] H. Li, "Learning to rank for information retrieval and natural language processing," *Synth. Lect. Hum. Lang. Technol.*, vol. 4, no. 1, pp. 1–113, Apr. 2011.

[12] S. Badsha, X. Yi, I. Khalil, D. Liu, S. Nepal, E. Bertino, and K.-Y. Lam, "Privacy preserving location-aware personalized Web service recommendations," *IEEE Trans. Services Comput.*, early access, May 22, 2018, doi: 10.1109/TSC.2018.2839587.

[13] J. Zhu, P. He, Z. Zheng, and M. R. Lyu, "A privacy-preserving QoS prediction framework for Web service recommendation," in *Proc. IEEE Int. Conf. Web Services*, Jun. 2015, pp. 241–248.

[14] T. R. Hoens, M. Blanton, A. Steele, and N. V. Chawla, "Reliable medical recommendation systems with patient privacy," *ACM Trans. Intell. Syst. Technol.*, vol. 4, no. 4, pp. 1–31, Sep. 2013.

[15] H. Polat and W. Du, "Privacy-preserving collaborative filtering using randomized perturbation techniques," in *Proc. 3rd IEEE Int. Conf. Data Mining*, Nov. 2003, pp. 625–628.

[16] V. Nikolaenko, S. Ioannidis, U. Weinsberg, M. Joye, N. Taft, and D. Boneh, "Privacy-preserving matrix factorization," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, 2013, pp. 801–812.

[17] A. Bilge, C. Kaleli, I. Yakut, I. Güneş, and H. Polat, "A survey of privacy-preserving collaborative filtering schemes," *Int. J. Softw. Eng. Knowl. Eng.*, vol. 23, no. 8, pp. 1085–1108, 2013.

[18] X. Zheng and Z. Cai, "Privacy-preserved data sharing towards multiple parties in industrial IoTs," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 5, pp. 968–979, May 2020.

[19] Y. Wang, J. Yu, B. Yan, G. Wang, and Z. Shan, "BSV-PAGS: Blockchain-based special vehicles priority access guarantee scheme," *Comput. Commun.*, vol. 161, pp. 28–40, Sep. 2020.

[20] S. Liu, J. Yu, Y. Xiao, Z. Wan, S. Wang, and B. Yan, "BC-SABE: Blockchain-aided searchable attribute-based encryption for cloud-IoT," *IEEE Internet Things J.*, vol. 7, no. 9, pp. 7851–7867, Sep. 2020.

[21] Z. Cai and X. Zheng, "A private and efficient mechanism for data uploading in smart cyber-physical systems," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 2, pp. 766–775, Apr. 2020.

[22] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2007, pp. 321–334.

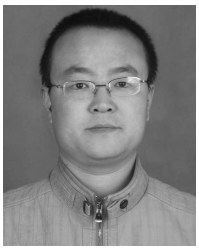
[23] S. Nakamoto and A. Bitcoin. (2008). *A Peer-to-Peer Electronic Cash System*. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[24] A. Gionis, P. Indyk, and R. Motwani, "Similarity search in high dimensions via hashing," *VLDB*, vol. 99, no. 6, pp. 518–529, 1999.

[25] B. Yan, J. Yu, Y. Wang, Q. Guo, B. Chai, and S. Liu, "Blockchain-based service recommendation supporting data sharing," in *Proc. Int. Conf. Wireless Algorithms, Syst., Appl.* Cham, Switzerland: Springer, 2020, pp. 580–589.

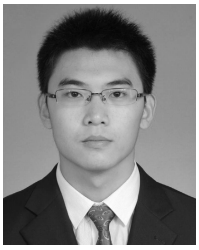
[26] C. Yan, X. Cui, L. Qi, X. Xu, and X. Zhang, "Privacy-aware data publishing and integration for collaborative service recommendation," *IEEE Access*, vol. 6, pp. 43021–43028, 2018.

BIWEI YAN received the B.S. and M.E. degrees in computer science from Qufu Normal University, China, in 2015 and 2017, respectively, where she is currently pursuing the Ph.D. degree with the School of Mathematical Sciences. Her research interests include information security and privacy, blockchain, and machine learning.



ANMING DONG received the B.E. degree in electronic information science and technology from Liaocheng University, Liaocheng, China, in 2004, the M.E. degree in communications and information systems from Lanzhou University, Lanzhou, China, in 2007, and the Ph.D. degree in communications and information systems from Shandong University, Jinan, China, in 2016. He is currently an Associate Professor with the School of Computer Science and Technology, Qilu University of Technology (Shandong Academy of Sciences), Jinan, China. His research interests include wireless communications, wireless networks, signal processing, and machine learning for wireless systems. He was a recipient of the Excellent Doctoral Dissertation Awards of Shandong Province, in 2017.

BAOBAO CHAI received the B.S. degree in medical information engineering from Shandong First Medical University (Shandong Academy of Medical Sciences), China, in 2018. He is currently pursuing the M.S. degree with the School of Computer Science and Technology, Qilu University of Technology (Shandong Academy of Sciences), China. His main research interests include industrial IoT, access control, and blockchain.



YUBING HAN (Member, IEEE) received the Ph.D. degree from the School of Engineering, China Agricultural University, in 2017. He became a Lecturer with the School of Computer Science, Qilu University of Technology, China, in 2017. His main research interests include smart city, wireless networking, and automotive electronics. He is particularly interested in designing and analyzing circuit design in engineering application. He is a member of ACM and the China Computer Federation (CCF).

GUANGLIN ZHOU received the B.E. degree in computer science and technology from Zaozhuang University, China, in 2019. He is currently pursuing the master degree with the School of Computer Science and Technology, Qilu University of Technology (Shandong Academy of Sciences), China. His research interests include security and privacy, blockchain, and federated learning.

FANGXIN ZHAO received the B.E. degree in computer science and technology from Zaozhuang University, China, in 2019. He is currently pursuing the master degree with the School of Computer Science and Technology, Qilu University of Technology (Shandong Academy of Sciences), China. His research interests include blockchain and federated learning.