Received August 5, 2020, accepted August 12, 2020, date of publication August 17, 2020, date of current version August 27, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3016986

An Efficient Blockchain-Based Bidirectional Friends Matching Scheme in Social Networks

FEIHONG YANG[1,2], YING WANG[1,2], CHUNLEI FU[1,2], CHUNQIANG HU[1,2], (Member, IEEE), AND ARWA ALRAWAIS[3], (Member, IEEE)

[1]School of Big Data and Software Engineering, Chongqing University, Chongqing 400044, China
[2]Key Laboratory of Dependable Service Computing in Cyber Physical Society, Ministry of Education, Chongqing University, Chongqing 400044, China
[3]College of Computer Engineering and Sciences, Prince Sattam Bin Abdulaziz University, Al-Kharj 11942, Saudi Arabia

Corresponding authors: Chunlei Fu (clfu@cqu.edu.cn) and Chunqiang Hu (hcq0394@gmail.com)

This work was supported in part by the National Key Research and Development Project under Grant 2018YFB2101202, in part by the Fundamental Research Funds for the Central Universities under Grant 2019CDQYRJ006, in part by the National Natural Science Foundation of China under Grant 61702062, in part by the Chongqing Research Program of Basic Research and Frontier Technology under Grant cstc2018jcyjAX0334, in part by the Key Project of Technology Innovation and Application Development of Chongqing under Grant cstc2019jscx-mbdxX0044 and Grant cstc2019jscx-mbdxX0020, in part by the Overseas Returnees Innovation and Entrepreneurship Support Program of Chongqing under Grant cx2018015, and in part by the Deanship of Scientific Research at Prince Sattam Bin Abdulaziz University through the Research Project under Grant 2019/01/1041.

ABSTRACT In social networks, the personal attributes or hobbies of the users are exposed to the server to establish the relationships. Service providers may store these information for commercial purpose or statistical analysis. Furthermore, the server may expose to external attacks, which may disclose users' privacy information. In this paper, we present a hierarchical blockchain-based attribute matching scheme, which realizes privacy-preserving attribute matching under multiple semi-trusted servers. The scheme employs CP-ABE and bloom filter to satisfy the requirements of the users to make friend discovery, and reduces the computation cost of users by outsourcing decryption of CP-ABE. Besides, the hierarchical blockchain only implements the consensus and storage of matching results on the blockchain, while the complex calculations and a large amount of data storage are off-chain, which reduces the consumption of the blockchain and improves the operation efficiency. Finally, we prove the scheme can resist single point failure, collusion attack, internal attack and external attack, the experimental results demonstrate the proposed scheme is feasibility and efficiency.

INDEX TERMS Attribute matching, friend discovery, CP-ABE, outsourcing decryption, hierarchical blockchain, social network.

I. INTRODUCTION

Social social networking provides an online platform to people to build social relationships with others, who have similar personal attributes such as age, home address, education background, etc. However, there is a risk of privacy leakage. Social network platforms may use user attributes for statistical, advertising or profit-making purposes [1], [2]. Such behavior will compromise users' privacy, which affect users' real life [3], [4]. For example, more than 25 gigabytes user information of an extramarital affair platform named Ashley Madison was leaked, including real name, home address and other information, and many users feared being publicly shamed [5], [6].

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

The basic idea of friend discovery in social networks is to compare the similarity of attributes between two users without leaking users' information. There are three categories to address the problem. The first category uses a set of attributes to summarize user's information, and employs Private Set Intersection(PSI) or Private Set Intersection Cardinality(PSI-CA) to execute attribute matching [5], [7]–[9]. The second category employs vectors to represent user's information, and the vector distance is calculated by dot product calculation to represent social distance [10]–[12]. The third category takes advantage of Ciphertext Policy Attribute-Based Encryption(CP-ABE) and access control to achieve friend discovery [13]–[18]. Nevertheless, there are some issues in existing schemes:

- Secure Multiparty Computing(SMC), homomorphic encryption or vector-based dot product calculation

are employed to execute the friend discovery. But these methods either consume massive computing resources or are vulnerable to statistical analysis attacks.
- The schemes just considers the requirements of the initiator rather than the needs of both, which is not in line with the actual situation of making friends.
- The existing schemes do not consider that social network platforms collude with some users to steal the privacy of others.

Additionally, most of the above solutions are based on point-to-point implementation. The server can reduce the computation cost of users during friend discovery, but the server is semi-trusted [19]. Blockchain has attracted much attention from scholars because of its characteristics of decentralized, immutable and traceable. In recent years, it has been widely used in Internet of Thing, social network, crowdsourcing, vehicle network and other fields [20]–[24].

To solve the problems of friend matching in social network, we propose an efficient and privacy-preserving friend matching based on blockchain in social networks. The contributions of the paper are summarized as follows:

- We proposed an attribute matching mechanism based on the hierarchical blockchain and outsourcing CP-ABE for friend discovery in social networks, which can achieve the attribute matching in semi-honest social network platforms and reduce computing consumption of users.
- The scheme satisfies the needs of the users by using CP-ABE and bloom filter, which is more appropriate to the actual situation of making friends.
- A novel blockchain architecture is proposed to achieve the decentralization and auditability of friend matching in social network, which reduce storage consumption of the blockchain.

The rest of this paper is organized as follows. Section II discusses related work. Section III provides detailed system model, new blockchain architecture and threat model. The preliminaries are introduced in Section IV. And the proposed scheme is elaborated in Section V. Section VI and Section VII provides security analysis and performance evaluation for our scheme, respectively. In Section VIII, we summarize our research and future work.

II. RELATED WORKS

The most existing schemes consider users' profiles as sets of attribute and measure similarity by calculating the intersection of attributes. The Private Set Intersection(PSI) technique for achieving attribute matching was first proposed by Li *et al.* [7], which is based on Secure Multi-party Computing(SMC). Yi *et al.* [5] proposed a profiles matching scheme based on homomorphic encryption in multiple social networks, which provides profile privacy-preserving. The basic idea is to judge whether the dissimilarity of two users is less than the threshold given by the user.

The other methods regards user's attributes as a vector, and the server calculates the dot product of two attribute vectors to obtain their similarity. Gao *et al.* [11] presented a multiple keys profile-matching protocol based on additive homomorphism to calculate the dot product of two vectors. Then, some dot product schemes were proposed which abandon homomorphic encryption and have lower computing costs. Luo *et al.* [10] set weight for each attribute, i.e., the attribute set is represented as a matrix, and then used a lightweight confusion matrix transformation algorithm to protect user information. And Li *et al.* [25] mixed the each attribute vector with random noise to realize attribute matching.

CP-ABE has been widely used in social networks [26]–[28]. Bethencourt *et al.* [29] first proposed CP-ABE in 2007. Waters [30] proposed a more efficient implementation of CP-ABE. Luo *et al.* [31] designed a friend discovery architecture based on CP-ABE and multiple attributed authority, which uses Shamir's scheme to store the master key distributedly. Li *et al.* [32] proposed a point-to-point pre-matching scheme, using Bloom filter to reduce the computational load of users performing decryption of CP-ABE, and they elaborated how to establish a verifiable secure communication channel between matched users. Based on this, Cui *et al.* [16] designed a receiver anonymous attribute matching scheme using CP-ABE and bloom filter. Qi *et al.* [15] combined searchable encryption with CP-ABE and proposed a friend discovery protocol with hidden attributes and fine-grained access control.

Considering the unreliability of the centralized server, the researchers applied the blockchain technology to social networks architecture. Jiang and Zhang [33] and Gu *et al.* [34] proposed social network architectures that use the blockchain and smart contracts instead of centralized servers to provide social networking services. They mainly take advantage of the immutability of traditional blockchain and the fairness of smart contracts.

III. SYSTEM OVERVIEW AND THREAT MODEL

In this section, we formalize the system model, the hierarchical blockchain architecture and the threat model.

A. SYSTEM MODEL

As depicted in Figure 1, the proposed system model consists of five entities, Trusted Authority(*TA*), Social Networking Platform(*SNP*), Users, Consortium Blockchain(*BC*), Proxy Cloud Computing Server(*PCCS*).

- ***TA***: A trusted third party is responsible for key generation.
- ***SNP***: It's a social networking platform with storage and computing capabilities such as Twitter and Facebook. It is the consensus node in the blockchain, which is called the miner in traditional blockchain networks. Under the chain, it provides social networking services and information storage services.
- ***Users***: The users would like to find friends in social networks, which has the similarly attributes. The initiator

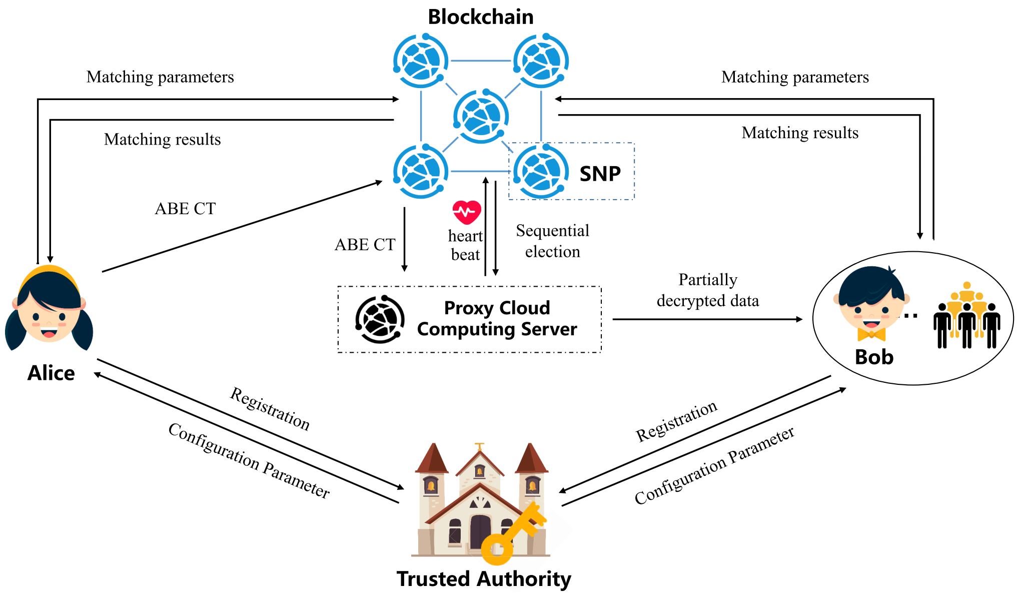


FIGURE 1. System architecture.

Alice and the responder Bob are two users, who want to establish the social relationship each other.

- ***BC***: *SNPs* constitute the blockchain consensus nodes. Users have blockchain accounts, which do not participate in the consensus and storage. Smart contract on *BC* determines whether two users can become friends.
- ***PCCS***: *PCCS* is selected from *SNPs* periodically via smart contract. It is responsible for partial decryption of CP-ABE to reduce users' computation consumption.

In the system, each user has two attribute sets S and R, S is representing his/her private attributes, and R is representing his/her requirements for making friends. To solve that people have same attributes which describe them differently, for example, Alice and Bob both like to sing, but Alice inputs "sing" and Bob inputs "I like to sing". Therefore, we provide attribute space A. The elements in S and R are the hash values of the elements selected from A, i.e.,

$$S = \{s_1, \ldots, s_n\}, R = \{r_1, \ldots, r_m\},$$
$$\forall s_i, r_j \in A\{h(att_1), \ldots, h(att_z)\}$$

In our scheme, friend matching in the social network should meet the following two conditions.

1) The percentage of attribute similarity between two users is greater or equal to a certain threshold given by the user.
2) Both parties' attributes meet each other's requirements.

In addition, *SNPs* do not anything about the users' attribute sets during the process of friend matching.

B. HIERARCHICAL BLOCKCHAIN ARCHITECTURE

One of our major contributions is proposed a hierarchical blockchain architecture. The performance of blockchain can be obviously improved by the hierarchical design [35]. Figure 2 shows the consortium blockchain architecture.

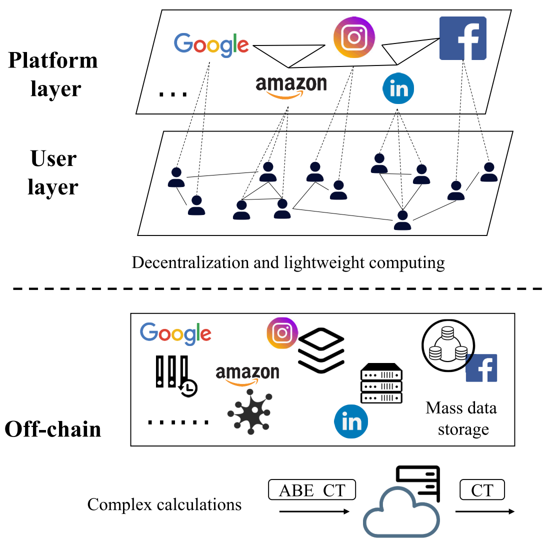


FIGURE 2. Hierarchical blockchain architecture.

The architecture divides the consortium blockchain into three parts: platform layer, user layer and off-chain.

The platform layer is composed of various *SNPs* and performs friend matching by smart contracts. All *SNPs* are consensus on the matching results and these records are stored in the blockchain. And smart contract periodically select a *PCCS* from *SNPs* to perform outsourcing encryption of CP-ABE.

A large number of users make up the user layer, which allows users to communicate casually with others, and they can make friends who meet specific attributes via the platform layer.

Off-chain stores the data that means each platform has private databases to store users' information. It indicates the blockchain stores matching records instead of all the users' information to reduce storage consumption of each consensus node. Additionally, to improve the efficiency of smart contracts, complex cryptographic calculations are also under off-chain. Each operation on the blockchain is lightweight and low storage consumption which can effectively solve the issues of low efficiency of smart contract execution and large storage burden in the blockchain.

C. THREAT MODEL

In our system, we consider the following threat models:

- **Single Point Failure**: Some *SNP*s' servers may be failure and stop providing services, or even be hijacked by attackers, which may cause system paralysis or users' information leakage.
- **External Attack**: The communication of each entity in the system may be subject to external attacks, such as replay attack, tampering attack, etc.
- **Internal Attack**: Assume that entity is honest but curious, but they want to infer the users' attribute information from the obtained data. Besides, the *SNP* may give the wrong execution result and cause the match to fail.
- **Collusion Attack**: *SNP*s may collude with some users to obtain other users' personal information.

Based on the above threat model, our system design goals are listed as follows:

- **Data Confidentiality**: Data privacy-preservation is the main purpose of the scheme. The users' attribute information should not be obtained by anyone under the above attacks.
- **Data Integrity**: The data integrity should be provided in the communication between entities.
- **Low Storage Cost for Blockchain**: The data in the blockchain is constantly added and cannot be deleted, and the data occupies the same amount of storage space of all consensus nodes. That is, if there are k consensus nodes, the storage consumption is k times of the traditional scheme, so the amount of data on the blockchain should be as small as possible.
- **Low Computation Cost for Blockchain**: Since the smart contract cannot be modified once it is deployed, and it is executed by all nodes, the algorithm in the smart contract should be simple, and the computation cost should be as low as possible.
- **Low Computation Cost for Users**: Due to the low computing power of users, the system should have the low computation cost on the users' side.

IV. PRELIMINARIES

In this section, we introduce the related cryptographic technique and background knowledge.

A. BILINEAR MAPS

We define $\mathbb{G}_1$ as a multiplicative cyclic groups of prime order p, and the generator of $\mathbb{G}_1$ is g. Let a bilinear map, $e : \mathbb{G}_1 \times \mathbb{G}_1 \longrightarrow \mathbb{G}_T$. And it has the following properties:

- **Bilinear**: for all $x, y \in \mathbb{G}_1$ and $a, b \in \mathbb{Z}_p$, we have $e(x^a, y^b) = e(x, y)^{ab}$.
- **Non-degeneracy**: $e(g, g) \neq 1$.
- **Computability**: for all $x, y \in \mathbb{G}_1$, there exists an algorithm to compute $e(x, y)$.

B. LINEAR SECRET SHARING SCHEMES(LSSS)

A secret sharing scheme Π in multiple entities $\mathcal{P}$ is called linear if:

- The shares for each entity form a vector based on $\mathbb{Z}_p$.
- The share-generating matrix M for scheme Π has ℓ rows and n columns. And there is a function ρ that associates the ith row of M to a party $\rho(i)$ and i is from 1 to ℓ. If we want share a secret value $s \in \mathbb{Z}_p$, we choose random numbers $r_2, .., r_n \in \mathbb{Z}_p$ to generate a vector $\vec{v} = (s, r_2, \ldots, r_n)$. Then we can get the vector $\overrightarrow{\mathrm{Mv}}$ with ℓ shares of the secret value s according to scheme Π. That is, share $(\overrightarrow{\mathrm{Mv}})_i$ belongs to party $\rho(i)$.

The secret s can be recovered as follows: Suppose that Π is an LSSS according to access structure $\mathbb{A}$. Define the authorized set S, and let $I \subset \{1, 2, \ldots, \ell\}$ be defined as $I = \{i : \rho(i) \in S\}$. Then, there exist constants $\{\omega_i \in \mathbb{Z}_p\}_{i \in I}$ such that, if $\{\lambda_i\}$ are valid shares of s according to scheme Π. Then s recovered as $s = \sum_{i \in I} \omega_i \lambda_i$. And s cannot be obtained for the unauthorized set $S \notin \mathbb{A}$.

C. CIPHERTEXT POLICY ATTRIBUTE-BASED ENCRYPTION

Ciphertext Policy Attribute-Based Encryption (CP-ABE) is typical asymmetric encryption method which can provide fine-grained access control [29]. In CP-ABE, there is one trusted authority who is responsible for generation of a public key PK and a master key MK. The user can encrypt data based on PK and the designed access structure $\mathbb{A}$. If the attribute sets of some users satisfy the above $\mathbb{A}$, they can decrypt it with their private keys which are generated by trusted authority based on their attribute sets and MK.

D. BLOCKCHAIN

As the critical technology of Bitcoin, blockchain has the characteristics of distributed, immutable and traceable [36], [37]. The blockchain composed of many nodes, maintaining the same database. There are mainly four concepts in the blockchain as shown below:

- **Transaction**: It is an operation that changes the state of the blockchain.
- **Block**: A block records a certain amount of transactions over a period of time. It represents the state change of the blockchain during this time.
- **Consensus**: All nodes verify whether a transaction is valid and then reach an agreement.
- **Chain**: Each block stores both the hash value of the previous block and the hash value of the current block. By this means, each block has a close association with the previous block and the latter block and form a chain.

E. SMART CONTRACT

Smart contract is an immutable program running on the blockchain, which is called "chaincode" in Fabric, for consistency we use "smart contract" to represent chaincode in the paper. Once the smart contract is invoked, it is executed by a consensus node, and the execution result will be checked by all consensus nodes. After being successful deployed, the smart contract is not controlled by anyone, and is viewed by everyone via a unique address. Therefore, smart contract can be trusted by users, thereby supersede the traditional central server.

V. THE PROPOSED SCHEME

In this section, we present the proposed scheme. Table 1 illustrate the frequently used notations in the scheme.

TABLE 1. The notations frequently used in the scheme.

Notations	Description
α, β, z, t	Random numbers.
MK, PK	The master key and the public key of CP-ABE.
SK, TK	User's secret key and transformation key of CP-ABE.
K_a	The symmetric secret key generated by Alice.
S, S'	User's private attribute set and the corresponding ciphertext.
R, R'	User's requirements for new friends and its ciphertext.
P_{sc}	The unique address of the smart contract.
P_a, P_b	Users' contact address in the social network.
PU_{bc}	The public key of the platform layer in the blockchain.
PR_{bc}	The private key of the platform layer in the blockchain.
CT_{abc}	The ciphertext (from Alice to blockchain).
CT_{apc}	The ciphertext (from Alice to *PCCS*).
CT_{pcb}	The ciphertext (from *PCCS* to Bob).
CT_{bsc}	The ciphertext (from Bob to the smart contract).

A. SYSTEM INITIALIZATION

In the system initiation phase, *TA* generates various keys for the system, and transmits them to the corresponding entities via secure channel. The asymmetrical key (PU_{bc}, PR_{bc}) are generated for blockchain consensus nodes to communicate with users.

Then *TA* generates the master key *MK* and the public key *PK*. Transformation key *TK* and secret key *SK* are generated for each user based on their attribute sets S. The outsourcing decryption of CP-ABE is constructed based on scheme [38]. In addition, the smart contract for periodically selecting *PCCS* from *SNP*s is deployed on the blockchain.

1) CP-ABE INITIALIZATION

TA selects a security parameter λ and generates a universe description $U = \{0, 1\}^*$. Then it chooses a bilinear group $\mathbb{G}_1$ of prime order p, a generator g. *TA* then selects two random numbers $\alpha, \beta \in \mathbb{Z}_p$. Besides, it needs to choose a hash function F that maps U to $\mathbb{G}_1$. After that, the public key is published:

$$PK = \{g, g^{\beta}, e(g, g)^{\alpha}, F\} \tag{1}$$

TA secretly stores the master key $MK = (PK, g^{\alpha})$ and exposes *PK* to *SNP*s and users, respectively.

2) USER REGISTRATION

The system sets the attribute space A that contains a large amount of attribute information such as gender, income, age, sports preference, education background, etc. When a new user joins the system, he/she first selects his/her attribute sets S and R from the attribute space A, then uploads S to *TA* to get transformation key *TK* and secret key *SK*.

TA picks a random number $t' \in \mathbb{Z}_p$ to create a temp key as:

$$K' = g^{\alpha} g^{\beta t'}, L' = g^{t'}, \quad \forall x \in S\ K'_x = F(x)^{t'} \tag{2}$$

Then it chooses another random number $z \in \mathbb{Z}_p$ and sets $t = t'/z$. Then it generates the transformation key *TK* as:

$$\begin{aligned} PK, K &= K'^{1/z} = g^{(\alpha/z)} g^{\beta(t'/z)} = g^{(\alpha/z)} g^{\beta t}, \\ L &= L'^{1/z} = g^{t'/z} = g^{t}, \{K_x\}_{x \in S} = \{K_x'^{1/z}\}_{x \in S} \end{aligned} \tag{3}$$

The *TK* is sent to the *PCCS*. The private key $SK = \{z, TK\}$ is sent to the user. Noted that when the user's personal attribute set changes, *TK* and *SK* should be regenerated.

3) SMART CONTRACT DEPLOYMENT

In the system initialization phase, the rules for electing *PCCS* need to be written into the smart contract. The *PCCS* is periodically elected from *SNP*s. The election algorithm is depicted in Figure 3. The smart contract containing the election algorithm is deployed on the blockchain, and all *SNP*s comply with the smart contract.

B. USERS COMMUNICATION

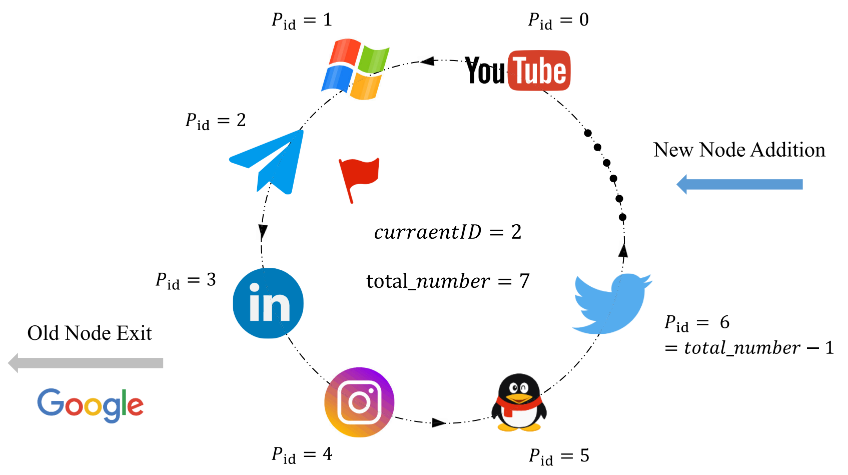
As demonstrated in Figure 4, when the initiator Alice wants to make friend discovery, she needs to execute three steps: matching preparation, smart contract initialization and data upload. The blockchain first checks the data. Then *PCCS* partially decrypts the data, and send it to the users who meet the requirement. If the receiver Bob also wants to execute friend discovery, he performs two steps: simple decryption and smart contract invocation.

1) MATCHING PREPARATION

In the preparation phase of matching, Alice first uses symmetric key K_a to encrypt each element of her private attribute set S_a to get S'_a.

$$S'_a = \{s'_{a1}, \ldots, s'_{an} \mid \forall s_{ai} \in S_a, s'_{ai} = E_{K_a}(s_{ai})\} \tag{4}$$

Then she encrypts her social network address P_a, $P'_a = E_{K_a}(P_a)$. Additionally, Alice constructs the Bloomer filter. She first initializes an array B of m bits with 0, and selects k hash functions $\{h_1, h_2, \ldots, h_k\}$. Alice uses these k hash functions to hash each value of S'_a. The hash value of s'_{ai} is its position in the array B, and the value of this bit in array B is set to 1. Figure 5 shows the procedure of Bloom filter when $k = 3$.



Each *SNP* has a unique serial number **Pid**, and the serial number increases monotonically from 0.
total_number represents the current total number of *SNP*s.
currentID is the serial number of the current *PCCS*.

- **Server Election**. The function is automatically executed when one of the following three conditions is met. First, the current *PCCS*'s term ends. Second, other nodes do not receive the heartbeat from *PCCS* within T_{ss}. Third, other nodes do not receive the CT_{bsc} from any users within T_{ss}.
  $\{currentID = (currentID + 1) \bmod total\_number;\}$
- **New Node Addition**. If a new *SNP* joins the consortium blockchain. The function will be executed automatically.
  $\{Pid = total\_number\ ;$
  $total\_number = total\_number + 1\}$
- **Old Node Exit**. If an *SNP* exits from the consortium blockchain. Pid' represents the sequence number of the *SNP*. The function will be executed automatically.
  $\{total\_number = total\_number - 1\ ;$
  if $Pid' == currentID$: function "Server Eelection" is invoked;
  if $Pid' < currentID$: $currentID = currentID - 1$;
  All serial numbers larger than Pid' are subtracted by one;}

FIGURE 3. Smart contract for proxy server election.

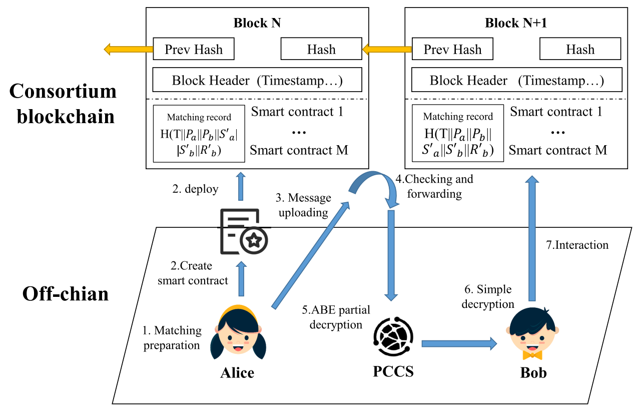


FIGURE 4. Users communication.

2) SMART CONTRACT INITIALIZATION

Alice initiates a transaction that creates a smart contract for friend matching. The smart contract is created here but not performed, so we elaborate its details in Section V-C. Note that the smart contract indicates that Alice is looking for friends. Alice needs to upload the following data to the platform layer including the instruction I, the ciphertext of the attribute set S'_a, the Bloom filter B, the ciphertext of the contact address P'_a, and the number of friends *max_num*.

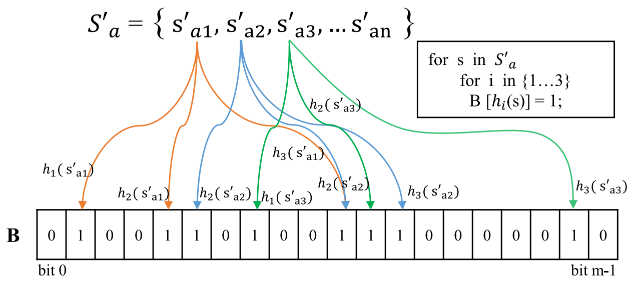


FIGURE 5. k = 3, the generation process of bloomer filter.

Hence, the data transmitted from Alice to the platform layer to create smart contract is as follows:

$$\begin{aligned} C_{abc} &= \{I \| T \| S'_a \| P'_a \| B \| max\_num \| \{h_1..h_k\}\}, \\ CT_{abc} &= E_{PU_{bc}}(C_{abc} \| H(C_{abc})) \end{aligned} \tag{5}$$

After the smart contract is successfully deployed, Alice will get a unique address P_{sc}. other users invoke the smart contract via P_{sc}.

3) DATA UPLOAD

The data is uploaded to the *PCCS* including three components: P_{sc}, K_a and $\mathcal{M}$. P_{sc} is the access address of smart contract, K_a is the symmetric secret key of AES, $\mathcal{M}$ is Alice's self-introduction or any other information that allows Bob to make a preliminary decision whether he is willing to make friends with Alice. So the data needs to be encrypted by CP-ABE as $\mathcal{M}' = \mathcal{M}\|K_a\|P_{sc}$.

Then Alice converts the R_a into LSSS access structure (M, ρ). M is an $\ell \times n$ matrix, and ρ is a function, in which the rows of M is associated with attributes. Alice first generates a random vector $\vec{v} = (s, y_2, \dots, y_n) \in \mathbb{Z}_p^n$, which is used to share the encryption s. For $i = 1$ to ℓ, $\lambda_i = \vec{v} \cdot M_i$, and the ith row of M is the vector Mi. Besides, Alice generates another ℓ a random sequences $\{r_1, \dots r_\ell \in \mathbb{Z}_p\}$. Then the ciphertext CT_{ab} is calculated as

$$\begin{aligned} C &= \mathcal{M}' e(g,g)^{\alpha s}, C' = g^s, \\ &(C_1 = g^{\beta\lambda_1} \cdot H(\rho(1))^{-r_1}, D_1 = g^{r_1}, \dots, \\ &(C_\ell = g^{\beta\lambda_\ell} \cdot H(\rho(\ell))^{-r_\ell}, D_\ell = g^{r_\ell}) \end{aligned} \tag{6}$$

along with a description of (M, ρ). Then Alice sends CT_{apc} to the platform layer instead of directly to *PCCS*.

$$\begin{aligned} C_{apc} &= \{T\|CT_{ab}\|max\_number\}, \\ CT_{apc} &= E_{PU_{bc}}(C_{apc}\|H(C_{apc})) \end{aligned} \tag{7}$$

4) DATA VERIFICATION

After receiving message from Alice, *SNP*s in the platform layer decrypts CT_{apc} with PR_{bc} to get C_{apc} and check data integrity and availability with T and hash values.

Then, the blockchain checks whether the data is legal, i.e. whether there are symmetric secret key or plaintext of access policies in CT_{apc}.

After verification, the current *PCCS* performs partial decryption of CP-ABE offline.

5) CP-ABE PARTIAL DECRYPTION

The *PCCS* uses users' transformation key TK to decrypt CT_{ab} in C_{apc} to get the partially decrypted data CT'_{ab}. If a user does not satisfy the requirements of Alice, that is, his/her S and TK does not meet the access structure(M, ρ), the decryption fails. Suppose that S meets the access structure and let $I \subset \{1, 2, 3, \dots, \ell\}$ be defined as $I = \{i : \rho(i) \in S\}$. Then, let $\{\omega_i \in \mathbb{Z}_p\}_{i\in I}$ be a set of constants such that if $\{\lambda_i\}$ are valid shares of any secret s according to M, then s can be computed as $\sum_{i\in I} \omega_i\lambda_i = s$. Therefore, the *PCCS* can computes:

$$\begin{aligned} &e(C', K)/(e(\prod_{i\in I} C_i^{\omega_i}, L) \cdot \prod_{i\in I} e(D_i^{\omega_i}, K_{\rho(i)})) \\ &= e(g,g)^{s\alpha/z} e(g,g)^{\beta st}/(\prod_{i\in I} e(g,g)^{t\beta\lambda_i\omega_i}) \\ &= e(g,g)^{s\alpha/z} \end{aligned} \tag{8}$$

Then the *PCCS* obtains partially decrypted data $CT'_{ab} = (C, e(g,g)^{s\alpha/z})$. It sends CT_{pcb} to Bob.

$$C_{pcb} = \{T\|CT'_{ab}\}, \quad CT_{pcb} = E_{PR_{bc}}(C_{pcb}\|H(C_{pcb})) \tag{9}$$

The *PCCS* will repeat the above process until Alice makes *max_num* new friends.

6) SIMPLE DECRYPTION

Bob checks T and the hash value to confirm whether the data has been replayed or modified. After that, he computes simple decryption:

$$C/(e(g,g)^{s\alpha/z})^z = \mathcal{M}' \cdot e(g,g)^{\alpha s}/e(g,g)^{s\alpha} = \mathcal{M}' \tag{10}$$

7) SMART CONTRACT INVOCATION

If Bob wants to make new friends after he obtains the message $\mathcal{M}$ sent by Alice, he should make sure whether Alice satisfies his requirements for making friends. Firstly, Bob encrypts his social network contact address P_b as $P'_b = E_{K_a}(P_b)$. Then he uses K_a to encrypt the attribute sets S_b and R_b to get S'_b and R'_b, respectively.

$$\begin{aligned} S'_b &= \{s'_{b1}, \dots, s'_{bn} \mid \forall s_{bi} \in S_b, s'_{bi} = E_{K_a}(s_{bi})\}, \\ R'_b &= \{r'_{b1}, \dots, r'_{bm} \mid \forall r_{bi} \in R_b, s'_{ri} = E_{K_a}(s_{ri})\} \end{aligned} \tag{11}$$

Furthermore, Bob sets the minimum percentage of attribute similarity between them to *min_percent*. Bob sends data to the platform layer as follows:

$$\begin{aligned} C_{bsc} &= \{T\|S'_b\|R'_b\|P'_b\|min\_percent), \\ CT_{bsc} &= E_{PU_{bc}}(C_{bsc}\|H(C_{bsc})) \end{aligned} \tag{12}$$

Bob initiates a transaction to send the CT_{bsc} to the smart contract via the address P_{sc}.

C. ATTRIBUTE MATCHING

After the transaction initiated by Bob is received, the consensus nodes first confirm the validity and the data integrity by T and hash value. After that, smart contract is executed by *SNP*, which mainly verifies whether Alice satisfies Bob's requirements via a bloom filter. If all elements of R'_b are in Bloom Filter B, it proves that Alice meets Bob's needs for making friends. Besides, the smart contract also needs to calculate whether the percentage of attribute similarity between Alice and Bob is greater or equals to the threshold required by Bob.

Algorithm 1 illustrates the attribute matching algorithm, which verifies whether Alice and Bob can be friends and matches up to *max_num* friends for Alice. If the match is successful, a communication channel will be established between Alice and Bob. In addition, because the smart contract will be invoked by multiple responders, bloom filter is more efficient than traversal S'_a and R'_b. The matching records will be stored in the blockchain. Note that we store the hash value of the input parameters instead of the original data, which can achieve matching transparency while protecting user privacy. If original parameters are stored

Algorithm 1 Attribute Matching Algorithm

```
Require:
    The ciphertext of users' social network contact addresses $P'_a, P'_b$.
    The ciphertext of users' attribute sets $S'_a, S'_b$.
    The Bloom filter generated by Alice $B$, and the corresponding hash functions $\{h_1, \ldots, h_k\}$.
    The maximum number of friends Alice wants to make, $max\_num$.
    The ciphertext of Bob's requirements for new friends $R'_b$.
    The minimum attribute similarity percentage set by Bob $min\_percent$.
1:  $matched\_num$ is used to record the number of friends Alice has matched
2:  if $max\_num \leq matched\_num$ then
3:      return $False$
4:  end if
5:  for $r$ in $R'_b$ do
6:      for ($i = 1;\ i \leq k;\ i{+}{+}$) do
7:          if $B[h_i(r)] == 0$ then
8:              return $False$
9:          end if
10:     end for
11: end for
12: Connect $S'_a$ and $S'_b$ to get $S$
13: Quicksort for $S$ to get $S'$
14: $same\_num = 0$
15: for ($i = 0; i < len(S'); i{+}{+}$) do
16:     if $S'[i] == S'[i+1]$ then
17:         $same\_num += 1$
18:     end if
19: end for
20: Calculate the number of attributes in $S'_b$ as $num\_b$
21: $same\_percent = same\_num/num\_b$
22: if $min\_percent > same\_percent$ then
23:     return $False$
24: else
25:     $matched\_num += 1$
26:     Send $P'_a$ and $P'_b$ to Bob and Alice respectively
27:     Hash values of all input parameters are stored in the blockchain
28:     return $True$
29: end if
```

directly, Alice or Bob can decrypt them with the symmetric secret key K_a to obtain other users' personal information. Moreover, matching records in the blockchain will not be deleted or modified by anyone, which can be managed by users.

VI. SECURITY ANALYSIS

In this section, we present the security of the proposed scheme under the threat model in Section III-C.

A. SINGLE POINT FAILURE

During the entire matching process, if an *SNP* loses response or is hijacked by malicious attackers, the system will not be affected and user attribute information will not be disclosed.

Challenge: *SNP* gets data CT_{abc}, CT_{apc} and CT_{bsc} from users. We Assume that an *SNP* server is broken down or hijacked by attackers. If the entire system crash or the attackers can get the user attribute information from CT_{abc}, CT_{apc} and CT_{bsc}, the attackers will have succeeded.

Proof:

- **The system will not crash**. Owning to the consensus mechanism, the blockchain will not be effected by one single *SNP* failure. For example, Practical Byzantine Fault Tolerance(PBFT) can hand up to 1/3 malicious consensus nodes [39]. Besides, if the *PCCS* crashes, as shown in Section V-A3, the smart contract will reelect a new *PCCS*.
- **User privacy will not be disclosed**. Assuming that the attacker can obtain blockchain node's public key PU_{bc} and private key PR_{bc}, it is also impossible to obtain any user information. Because as shown in Section VI-C, even the *SNP* cannot obtain user information.

B. EXTERNAL ATTACK

External attackers are entities that eavesdrop or tamper with data during the communication. Our scheme can protect user attribute information from various external attacks.

Challenge: The communication data in the system includes CT_{abc}, CT_{apc}, CT_{pcb} and CT_{bsc}. If external attackers can obtain users' information or change the matching result by replaying or tampering with these data, the attackers will have succeeded.

Proof: The data flow of the proposed scheme is depicted in Figure 4. The all data has been encrypted via the public key of the corresponding entity, so external attackers without corresponding private key cannot decrypt the transmitted data. Except that CT_{pcb} is encrypted by PR_{bc}, external attackers can use PU_{bc} to obtain C and CT'_{ab}. However, attackers are unable to tamper with CT'_{ab} and cannot get the users' information.

- **Attackers is unable to tamper with $\mathbf{CT'_{ab}}$**. Because the $CT_{pcb} = E_{PR_{bc}}(C_{pcb} \| H(C_{pcb}))$, which contains the hash value of the C_{pcb}. It means Bob can calculate the hash value of C_{pcb} and compare it with $H(C_{pcb})$. If they are not equal, it indicates the data has been tampered. Bob will destroy the data and report it to the platform layer.
- **Attackers cannot get the user information**. $CT'_{ab} = (C, e(g,g)^{s\alpha/z}), C = \mathcal{M}'e(g,g)^{\alpha s}$. And z is Bob's private key, so the attackers cannot decrypt C to get $\mathcal{M}'$ or any other information.

C. INTERNAL ATTACK

During the matching process, Alice, Bob, *PCSS*, *SNP* cannot know attribute information of the users.

Challenge: If internal attackers are able to obtain the user attribute information from the transmitted data, the attackers will have succeeded.

Proof: Firstly, during the matching process, Alice only receives the matching result, and Bob receives $CT_{pcb} = Enc(\mathcal{M}\|K_a\|P_{sc})$ and the matching result. Therefore, even if both parties decrypt the received data to get $\{\mathcal{M}, K_a, P_{sc}\}$, they can not obtain any attribute information of the other users. Secondly, the *PCCS* can only know partially decrypted data, because z is Bob's private key. Additionally, *SNP*s on the blockchain cannot learn users information, either. Because all attribute sets are encrypted by K_a, which is only known to Alice and Bob. Moreover, *SNP* can not learn anything from bloom filter B, because the attribute space A is large enough, and the *SNP* does not know the size of R_a.

D. COLLUSION ATTACK

The proposed protocol can resist collusion attacks from multiple entities, and we mainly analyze the collusion between *SNP*s and users.

Challenge 1: *SNP*s on the blockchain get S' and R' from Alice and bob, which is encrypted by K_a. We assume that a *SNP* and Alice collude to decrypt S'_b and R'_b with K_a (the same as Bob).That is, Alice adds K_a to the uploaded data. If they could obtain Bob's attribute set S_b, R_b, the collusion attack is successful.

Proof: Transactions in the blockchain are packaged into blocks by random consensus nodes. It means other *SNP*s may receive the symmetric key K_a from Alice, in which case Alice will be punished and the system will no longer provide friend matching for Alice. In addition, even in some consensus algorithms, the designated consensus node receives transactions. The transaction will be confirmed by all the consensus nodes and it will be detected if the transaction contains K_a.

Challenge 2: The *PCCS* gets CT_{apc} from Alice. We Assume that the current *PCCS* colludes with Alice. And Alice sends the plaintext of access control structure or the symmetric secret key to the *PCCS*. If they obtain Bob's attribute information, the collusion attack is successful.

Proof: As described in Section V-B4, before the *PCCS* gets the CT_{apc}, all *SNP*s have recorded the CT_{apc} and confirm whether the CT_{apc} is legal. If it contains K_a or the plaintext of the access control structure, Alice's operation is rejected and the match is terminated.

VII. PERFORMANCE EVALUATION

In this section, we first introduce the simulation experiment environment, then we analyze the performance of the scheme based on Hyperledger Fabric. Finally, the function of our scheme is compared with the existing schemes.

A. SIMULATION EXPERIMENT PREPARATION

The simulation experiment is implemented in C/C++ on the Ubuntu 16.06 virtual machine under Windows 10 with Inter Inter(R) Core(TM) i5-7500 CPU @ 3.40GHz. Our experiment focused on the implementation of AES, outsouring decryption of CP-ABE and consortium blockchain. We ignore the underlying communication consumption between entities.

Firstly, the traditional AES with 128-bit key was implemented in C and the bloom filter was constructed in C++ based on Open Bloom Filter library. Afterwards, the outsouring decryption of CP-ABE was conducted based on Stanford Pairing-Based Crypto (PBC) library. It should be noted that the main computational consumption in CP-ABE is caused by bilinear operation, so we ignore the arithmetic operation on $\mathbb{Z}_p$. In addition, we deployed a consortium blockchain on Hyperledger Fabric which is constructed based on Go and Docker. We set all *SNP*s as endorsing peers. And the smart contract was implemented in Go.

We choose Fabric instead of Ethereum, the reasons are listed.

- Ethereum is a public blockchain, which is almost impossible for *SNP*s to trust it and use it.
- The transaction in Ethereum requires an additional fee called ''Gas'', which users are unwilling to bear.
- Ethereum's consensus mechanism leads to low transaction execution efficiency, which does not apply to friend matching in social networks.

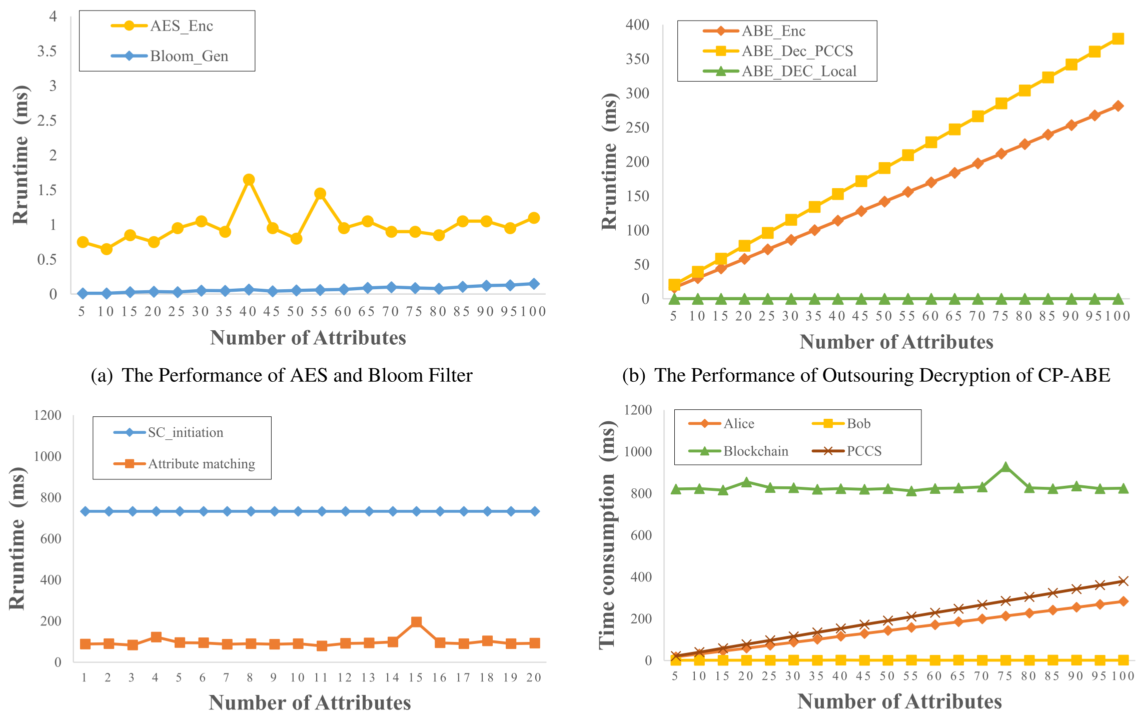
We focus on the relationship between the performance of the scheme and the number of attributes, so we assume that Alice and Bob have ℓ private attributes and ℓ preference attributes, respectively. And $\ell = \{5, 10, \ldots, 95, 100\}$. The experimental results are depicted in Figure 6, all the data were the average of 20 experiments.

B. PERFORMANCE ANALYSIS

Figure 6(a) shows the time consumed by AES encryption for different numbers of attributes. Each attribute in R and P is encrypted separately. It can be seen from the figure that the time consumption has not changed significantly with the increase of the number of attributes. Besides, we set the tolerable false positive probability of bloom filter as 0.0001. We can find that the bloom filter has extremely high efficiency, the time consumption is increasing with the number of attributes, but is not more than 0.2*ms*.

Figure 6(b) presents the performance of outsouring decryption of CP-ABE. We use E_G, E_T, P to denote the average time to complete an exponentiation in $\mathbb{G}_1$, exponentiation in $\mathbb{G}_T$ and a paring respectively. The average time they consume is 1.3953*ms*, 0.9876*ms*, and 0.1839*ms*. From the Section V, we know the encryption time consumption of CP-ABE is $\{P + E_T + (2\ell + 1)E_G\}$. And the partial decryption time consumption is $\{(2+\ell)P+2\ell E_G\}$. Most importantly, the local decryption time is E_T, which does not change with the number of attributes increasing.

Figure 6(c) depicts the relationship between the running time of the designed blockchain and the number of attributes, including the time that it takes to initiate a transaction to generate a smart contract, and the time that it takes to execute attribute matching. We record the full running time of



(a) The Performance of AES and Bloom Filter

(b) The Performance of Outsouring Decryption of CP-ABE

(c) The Performance of Blockchain

(d) Total Time Consumption of Each Entity

FIGURE 6. Performance evaluation of the scheme.

TABLE 2. Comparisons of functionality.

Scheme	Fine-Grained Matching	Low User Consumption	Bidirectional Matching	Collusion Resistance	Blockchain-Based
Ref. [31]	Y	N	N	N	N
Ref. [13]	Y	Y	Y	N	N
Ref. [10]	N	Y	N	⊥	N
Ref. [25]	N	Y	N	Y	N
Ref. [11]	Y	Y	N	Y	N
Ref. [32]	Y	Y	N	⊥	N
Our scheme	**Y**	**Y**	**Y**	**Y**	**Y**

the transaction from initiation to store in the blockchain. Compared with the time consumed by the blockchain, the time consumption caused by the changes in the number of attributes can be negligible.

The total time consumption of each entity is shown in Figure 6(d). Alice performs encryption of CP-ABE, encryption of AES and Bloom filter generation. For Bob, he needs to complete local decryption of CP-ABE and encryption of AES. For the blockchain, it performs smart contract initialization and attribute matching. For *PCCS*, it performs outsourcing decryption of CP-ABE. From the figure, the blockchain operation and outsourcing decryption of CP-ABE take the most time. It demonstrates the necessity of transferring complex calculations from the blockchain to the off-chain, which can significantly reduce the computation cost on the blockchain. What's more, an initiator will have multiple responders, so the reduction of time spent by the responder can effectively decrease the consumption of the entire matching process. The experimental result in Figure 6(d) demonstrates that Bob's time consumption is negligible, which proves the effectiveness of our scheme.

In summary, experimental results show that the scheme is effective and feasible.

C. FUNCTIONAL COMPARISON OF SIMILAR SCHEMES

We compare the proposed scheme to some similar advanced schemes introduced in Section II, and the comparison results are demonstrated in Table 2. All the schemes can achieve privacy preservation, so it is not displayed in the table. To achieve privacy preservation, there are five key factors are considered: Fine-Grained Matching, Low User Consumption, Bidirectional Matching, Collusion Resistance and Blockchain-Based. The Y in the table indicates that this feature is available in this scheme and the N represents that it is not. And $\perp$ means the scheme is a point-to-point attribute matching scheme, so collusion attack is not considered.

It can be concluded that we proposed a novel and complete friend matching protocol based on the new blockchain architecture with the above five characteristics.

VIII. CONCLUSION

In this paper, we present a privacy-preserving attribute matching scheme under multiple semi-trusted servers. In our scheme, we utilize CP-ABE and bloom filter to conduct bidirectional attribute matching and relieve the computation cost of users by outsourcing decryption. In addition, we design a novel hierarchical blockchain architecture, which massively reduces the storage consumption of the blockchain and improves operating efficiency. Security analysis and experiment results demonstrate that our scheme can resist single point failure attack, collusion attack, internal attack and external attack, and also provide effectively friend matching for users.

In the future, we consider using blockchain instead of the trusted third party to initialize CP-ABE, which is a challenge issue. In addition, we plan to analyze the security and efficiency of blockchain consensus and friend matching in practical application.

REFERENCES

[1] S. Oukemeni, H. Rifà-Pous, and J. M. M. Puig, "Privacy analysis on microblogging online social networks: A survey," *ACM Comput. Surv.*, vol. 52, no. 3, pp. 1–36, Jul. 2019.

[2] Y. Lin, Z. Cai, X. Wang, and F. Hao, "Incentive mechanisms for crowdblocking rumors in mobile social networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 9, pp. 9220–9232, Sep. 2019.

[3] Y. Pu, T. Xiang, C. Hu, A. Alrawais, and H. Yan, "An efficient blockchain-based privacy preserving scheme for vehicular social networks," *Inf. Sci.*, vol. 540, pp. 308–324, Nov. 2020.

[4] Z. Cai, Z. He, X. Guan, and Y. Li, "Collective data-sanitization for preventing sensitive information inference attacks in social networks," *IEEE Trans. Dependable Secure Comput.*, vol. 15, no. 4, pp. 577–590, Aug. 2018.

[5] X. Yi, E. Bertino, F.-Y. Rao, and A. Bouguettaya, "Practical privacy-preserving user profile matching in social networks," in *Proc. IEEE 32nd Int. Conf. Data Eng. (ICDE)*, May 2016, pp. 373–384.

[6] P. Chaudhary and B. B. Gupta, "A novel framework to alleviate dissemination of XSS worms in online social network (OSN) using view segregation," *Neural Netw. World*, vol. 27, no. 1, pp. 5–25, 2017.

[7] M. Li, N. Cao, S. Yu, and W. Lou, "FindU: Privacy-preserving personal profile matching in mobile social networks," in *Proc. IEEE INFOCOM*, Apr. 2011, pp. 2435–2443.

[8] Y. Ishikuro and K. Omote, "Privacy-preserving profile matching protocol considering conditions," in *Proc. Int. Conf. Netw. Syst. Secur.* Taipei, Taiwan: Springer, 2016, pp. 171–183.

[9] D. He, Z. Cao, X. Dong, and J. Shen, "User self-controllable profile matching for privacy-preserving mobile social networks," in *Proc. IEEE Int. Conf. Commun. Syst.*, Nov. 2014, pp. 248–252.

[10] E. Luo, Q. Liu, J. H. Abawajy, and G. Wang, "Privacy-preserving multi-hop profile-matching protocol for proximity mobile social networks," *Future Gener. Comput. Syst.*, vol. 68, pp. 222–233, Mar. 2017.

[11] C.-Z. Gao, Q. Cheng, X. Li, and S.-B. Xia, "Cloud-assisted privacy-preserving profile-matching scheme under multiple keys in mobile social network," *Cluster Comput.*, vol. 22, no. S1, pp. 1655–1663, Jan. 2019.

[12] G. Sheng, T. Wen, Q. Guo, and Y. Yin, "Privacy preserving inner product of vectors in cloud computing," *Int. J. Distrib. Sensor Netw.*, vol. 10, no. 5, May 2014, Art. no. 537252.

[13] T. Li, X. Li, and X. Liu, "An efficient privacy-preserving bidirectional friends matching scheme in mobile social networks," in *Proc. Int. Conf. Netw. Netw. Appl. (NaNA)*, Oct. 2019, pp. 51–57.

[14] W. Wang, F. Qi, X. Wu, and Z. Tang, "Distributed multi-authority attribute-based encryption scheme for friend discovery in mobile social networks," *Procedia Comput. Sci.*, vol. 80, pp. 617–626, 2016.

[15] F. Qi, X. Chang, Z. Tang, and W. Wang, "Searchable attribute-based encryption protocol with hidden keywords in cloud," in *Proc. Int. Conf. Dependability Sensor, Cloud, Big Data Syst. Appl.* Guangzhou, China: Springer, 2019, pp. 80–92.

[16] W. Cui, C. Du, and J. Chen, "CP-ABE based privacy-preserving user profile matching in mobile social networks," *PLoS ONE*, vol. 11, no. 6, Jun. 2016, Art. no. e0157933.

[17] T. Guo, K. Dong, L. Wang, M. Yang, and J. Luo, "Privacy preserving profile matching for social networks," in *Proc. 6th Int. Conf. Adv. Cloud Big Data (CBD)*, Aug. 2018, pp. 263–268.

[18] B. Chandrasekaran, Y. Nogami, and R. Balakrishnan, "An efficient hierarchical multi-authority attribute based encryption scheme for profile matching using a fast ate pairing in cloud environment," *J. Commun. Softw. Syst.*, vol. 14, no. 2, pp. 151–156, 2018.

[19] S. He, Q. Wu, X. Luo, Z. Liang, D. Li, H. Feng, H. Zheng, and Y. Li, "A social-network-based cryptocurrency wallet-management scheme," *IEEE Access*, vol. 6, pp. 7654–7663, 2018.

[20] S. Zhu, W. Li, H. Li, L. Tian, G. Luo, and Z. Cai, "Coin hopping attack in blockchain-based IoT," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4614–4626, Jun. 2019.

[21] Y. Pu, C. Hu, S. Deng, and A. Alrawais, "RPEDS: A recoverable and revocable privacy-preserving edge data sharing scheme," *IEEE Internet Things J.*, early access, May 26, 2020, doi: 10.1109/JIOT.2020.2997389.

[22] Z. He, Z. Cai, J. Yu, X. Wang, Y. Sun, and Y. Li, "Cost-efficient strategies for restraining rumor spreading in mobile social networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 3, pp. 2789–2800, Mar. 2017.

[23] S. Hasavari and Y. T. Song, "A secure and scalable data source for emergency medical care using blockchain technology," in *Proc. IEEE 17th Int. Conf. Softw. Eng. Res., Manage. Appl. (SERA)*, May 2019, pp. 71–75.

[24] S. Zhu, Z. Cai, H. Hu, Y. Li, and W. Li, "ZkCrowd: A hybrid blockchain-based crowdsourcing platform," *IEEE Trans. Ind. Informat.*, vol. 16, no. 6, pp. 4196–4205, Jun. 2020.

[25] R. Li, H. Li, X. Cheng, X. Zhou, K. Li, S. Wang, and R. Bie, "Perturbation-based private profile matching in social networks," *IEEE Access*, vol. 5, pp. 19720–19732, 2017.

[26] C. Li, Z. Zhang, and L. Zhang, "A novel authorization scheme for multimedia social networks under cloud storage method by using MA-CP-ABE," *Int. J. Cloud Appl. Comput.*, vol. 8, no. 3, pp. 32–47, Jul. 2018.

[27] Q. Huang, L. Wang, and Y. Yang, "Secure and privacy-preserving data sharing and collaboration in mobile healthcare social networks of smart cities," *Secur. Commun. Netw.*, vol. 2017, pp. 1–12, 2017.

[28] X. Sun, Y. Yao, Y. Xia, X. Liu, J. Chen, and Z. Wang, "Towards efficient sharing of encrypted data in cloud-based mobile social network," *KSII Trans. Internet Inf. Syst.*, vol. 10, no. 4, pp. 1892–1903, 2016.

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

[30] B. Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization," in *Proc. Int. Workshop Public Key Cryptogr.* Taormina, Italy: Springer, 2011, pp. 53–70.

[31] E. Luo, Q. Liu, and G. Wang, "Hierarchical multi-authority and attribute-based encryption friend discovery scheme in mobile social networks," *IEEE Commun. Lett.*, vol. 20, no. 9, pp. 1772–1775, Sep. 2016.

[32] H. Li, X. Cheng, K. Li, and Z. Tian, "Efficient customized privacy preserving friend discovery in mobile social networks," in *Proc. IEEE 35th Int. Conf. Distrib. Comput. Syst.*, Jun. 2015, pp. 225–234.

[33] L. Jiang and X. Zhang, "BCOSN: A blockchain-based decentralized online social network," *IEEE Trans. Comput. Social Syst.*, vol. 6, no. 6, pp. 1454–1466, Dec. 2019.

[34] K. Gu, L. Wang, and W. Jia, "Autonomous resource request transaction framework based on blockchain in social network," *IEEE Access*, vol. 7, pp. 43666–43678, 2019.

[35] C. Liu, Y. Xiao, V. Javangula, Q. Hu, S. Wang, and X. Cheng, "NormaChain: A blockchain-based normalized autonomous transaction settlement system for IoT-based E-commerce," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4680–4693, Jun. 2019.

[36] M. Crosby, P. Pattanayak, S. Verma, and V. Kalyanaraman, "Blockchain technology: Beyond bitcoin," *Appl. Innov.*, vol. 2, nos. 6–10, p. 71, 2016.

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

[38] M. Green, S. Hohenberger, and B. Waters, "Outsourcing the decryption of abe ciphertexts," in *Proc. USENIX Secur. Symp.*, no. 3, 2011, 2011.

[39] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, "An overview of blockchain technology: Architecture, consensus, and future trends," in *Proc. IEEE Int. Congr. Big Data (BigData Congress)*, Jun. 2017, pp. 557–564.

FEIHONG YANG received the B.S. degree from Chongqing University, China, in 2019, where he is currently pursuing the master's degree with the School of Big Data and Software Engineering. His research interests include the IoT applications based on blockchain, privacy, and security issues in blockchain.

YING WANG received the bachelor's degree in the Internet of Things from the Chongqing University of Technology. She is currently pursuing the master's degree with Chongqing University. Her major research interest includes information security.

CHUNLEI FU received the Ph.D. degree from the School of Automation, Chongqing University, China, in 2014, and the Ph.D. degree in software engineering from the School of Computer Science, Chongqing University. He is currently a Senior Engineer with the School of Big Data and Software Engineering, Chongqing University. His research interests include cybersecurity, blockchain, and software engineering.

CHUNQIANG HU (Member, IEEE) received the B.S. degree in computer science and technology from Southwest University, Chongqing, China, in 2006, the M.S. and Ph.D. degrees in computer science and technology from Chongqing University, Chongqing, in 2009 and 2013, respectively, and the Ph.D. degree in computer science from The George Washington University, Washington, DC, USA, in 2016. He was a Visiting Scholar with The George Washington University from 2011 to 2012. He is currently a Faculty Member with the School of Big Data and Software Engineering, Chongqing University. His research interests include privacy-aware computing, big data security and privacy, wireless and mobile security, applied cryptography, and algorithm design and analysis. He is a member of ACM. He was honored from the Hundred-Talent Program, Chongqing University.

ARWA ALRAWAIS (Member, IEEE) received the M.S. degree in computer science and the Ph.D. degree from the Department of Computer Science, The George Washington University, Washington, DC, USA, in 2011 and 2017, respectively. She holds a patent in system and method for remote authentication with dynamic usernames. Her current research interests include network security, wireless and mobile security, and algorithm design and analysis. She serves as a Professional Reviewer for several conferences and journals of the IEEE and ACM.