IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 14, 2019, accepted September 19, 2019, date of publication September 23, 2019,
date of current version October 4, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2943114

A Privacy-Preserving and Identity-Based Personalized Recommendation Scheme for Encrypted Tasks in Crowdsourcing

HUI YIN[1], YINQIAO XIONG[1,2], TIANTIAN DENG[1,2], HUA DENG[3], AND PEIDONG ZHU[1], (Senior Member, IEEE)

[1]College of Computer Engineering and Applied Mathematics, Changsha University, Changsha 410022, China
[2]College of Computer, National University of Defense Technology, Changsha 410073, China
[3]College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China

Correspsonding author: Yinqiao Xiong (yq.xiong@ccsu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61972058 and Grant 61902123, in part by the Science and Technology Key Projects of Hunan Province under Grant 2016JC2012, in part by the Natural Science Foundation of Hunan Province under Grant 2017JJ3371, and in part by the Outstanding Youth Research Project of Provincial Education Department of Hunan under Grant 17B030.

ABSTRACT Personalized task recommendation can guarantee that tasks are pushed to the right workers, and thus, requesters gain better-quality output from a crowdsourcing system. Requesters' tasks and workers' interests exposed to the remote crowdsourcing platform in the form of plaintext have raised serious privacy concerns. Encrypting tasks and interests is a feasible solution to protect the privacy of both the requesters and the workers. However, data encryption renders the existing crowdsourcing task recommendation techniques ineffective. To address this challenge, in this study, we transform the personalized task recommendation problem into a task access control and keyword-based search problem for encrypted tasks. On the basis of this idea, we first develop a new technique called *multi-authority attribute-based searchable encryption* by equipping the searchable capacity for Lewko and Waters's multi-authority attribute -based encryption. Then, we utilize the new technique to construct a secure and personalized recommendation scheme in crowdsourcing, which achieves accurate personalized task recommendation in a privacy-preserving manner by a seamless combination of attribute-based encryption and searchable encryption. We provide rigorous security proof and thorough security analysis for the proposed scheme. Extensive experiments demonstrate the correctness and practicality of the proposed scheme.

INDEX TERMS Attribute-based encryption, crowdsourcing, personalized task recommendation, privacy preservation, searchable encryption.

I. INTRODUCTION

A. MOTIVATION

The basic idea of crowdsourcing is to take full advantage of human intelligence to accomplish some complex tasks that cannot be effectively completed by the computer alone, such as imaging tagging, programming, and natural language processing [2]. Tasks are published to a crowdsourcing platform by the task requesters, through which authorized workers search for suitable tasks to perform to earn due rewards. The well-known Google MapMaker and Amazon MTurk are exactly such crowdsourcing platforms.

Obviously, it is a heavy burden for workers to locate appropriate tasks when facing a large number of diversiform tasks on crowdsourcing platforms. Staying online to pick tasks usually requires the workers to spend considerably more time than that required for completing them, which is likely to dampen the workers' enthusiasm of using crowdsourcing platforms. In contrast, if the workers casually choose tasks that do not match their professional skills and capabilities, poor-quality results will be returned, which discounts the practicality of the crowdsourcing system. Task recommendation can help workers to find the right tasks as well as help the requesters to receive good-quality output more quickly [3]. Recently, several excellent personalized crowdsourcing task recommendation schemes have been proposed to keep improving the recommendation accuracy [4].

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

In general, the crowdsourcing platform is operated by commercial organizations and is usually deployed in a cloud computing center. The task requesters and workers are allowed to access the platform through the Internet. As the cloud server cannot always be fully trusted [5], requesters can be reluctant to publish their tasks to the crowdsourcing platform in the plaintext form. A natural measure for preserving the privacy of tasks is to encrypt tasks before publishing them [6]. However, data encryption makes all of the existing task recommendation schemes in the plaintext environment ineffective. We need to develop a new technique to achieve task recommendation and privacy preservation simultaneously in a crowdsourcing system. To address this problem, Shu and Jia's [7] pioneering effort defined a secure crowdsourcing task recommendation model and leveraged searchable encryption [8] to propose a privacy-preserving crowdsourcing task recommendation scheme. In the scheme, different task requesters adopt different keys to encrypt their tasks and an authorized worker uses his/her own key to encrypt his/her interests; the crowdsourcing platform is responsible for matching the encrypted tasks with the encrypted interests and pushing the matched tasks to the worker. Therefore, the privacy of both the tasks and the workers is protected against the remote crowdsourcing platform.

Shu *et al.* works [7], [9]–[13] have provided effective approaches to achieve secure task recommendations in multiple user crowdsourcing environment. However, only matching between encrypted tasks and encrypted interests may not be sufficient for personalized task recommendations. For example, a freshman who is majoring in software engineering may be very interested in programming, yet he may not have sufficient skills to complete a complex programming task in time. If the task is recommended to the beginner, a bad outcome will be returned. To let the requesters receive a better-quality task output quickly, several personalized task recommendation schemes have been proposed on the basis of the worker's search history or interests [3], [14], [15]. However, performing such a personalized task recommendation is not a trivial task in the secure crowdsourcing task recommendation scenario, as both the requesters' tasks and the workers' search histories or interests are encrypted, which disables the existing recommendation models and recommendation algorithms.

In this paper, from a new perspective, we propose a privacy-preserving and identity-based personalized task recommendation scheme, by which an encrypted task can be recommended to the most suitable worker whose interests and identities match the task simultaneously, while the crowdsourcing platform knows nothing about the requester's task and the worker's interests. The main idea of the proposed scheme is that the requester defines a specified recommendation condition for a task and a worker whose identities satisfy the specified condition and whose interests match the task can obtain the task. Moreover, the entire task recommendation processes are conducted in the encrypted environment.

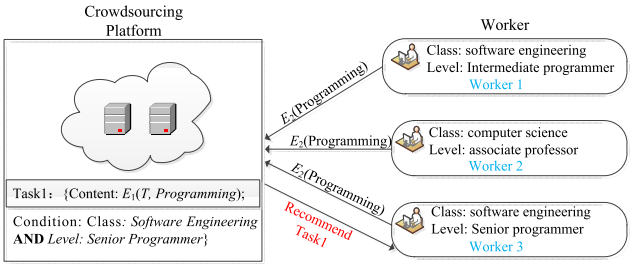


FIGURE 1. Example of the privacy-preserving and personalized task recommendation based on worker's attributes.

Obviously, the greatest challenge to achieve the identity-based *personalized* recommendation is how the crowdsourcing platform determines whether a worker's identities satisfy a task's recommendation condition in the ciphertext environment. To address the challenge, we transform the *personalization* into the problem that the requester controls who is qualified to accept a task by setting the access policy in the task. If and only if a worker's *identities* satisfy a task's access policy, the task could be recommended to the worker. If we use a set of descriptive attributes to denote a worker's identities, attribute-based encryption [16] may be a potential technique to solve this problem. However, the existing attribute-based encryption schemes cannot be directly applied to implement the task recommendation system because of the lack of keyword-based searchability, which makes the matching between the encrypted tasks and the encrypted interests unreachable. Therefore, in this study, we design a new approach called *multi-authority attribute-based searchable encryption* to achieve privacy-preserving and personalized task recommendations for crowdsourcing.

For the ease of understanding, we consider an example at the end of this subsection. Assume that a requester encrypts a programming task T with a Boolean formula: *"(Class: Software Engineering)* **AND** *(Level: Senior Programmer")* and then publishes the ciphertext to the crowdsourcing platform. The crowdsourcing platform only recommends the task to ones who are senior programmers and are majoring in software engineering if they submit their encrypted interest as "*Programming*", where *Software Engineering* and *Senior Programmer* are treated as a set of attributes describing a worker's identities.

Fig. 1 shows an example of the privacy-preserving and personalized task recommendation, where E_1 and E_2 are two encryption blocks used to encrypt the requester's tasks and the worker's interests, respectively.

B. CONTRIBUTIONS

Our contributions can be summarized as follows:

1. We present a *multi-authority attribute-based searchable encryption* by developing the state-of-the-art multi-authority attribute-based encryption in [1], which can simultaneously achieve fine-grained data access control and keyword based searching over encrypted data in a decentralization authority environment.

2. We design a privacy-preserving and identity-based personalized recommendation scheme for encrypted tasks in crowdsourcing by utilizing the proposed *multi-authority attribute-based searchable encryption*. The scheme allows the crowdsourcing platform to recommend a task to the most suitable workers based on the workers' identities without knowing any information about the requesters' tasks and the workers' interests.

3. We provide thorough security proof and analysis for the proposed scheme. The experimental evaluations show that the proposed scheme is correct and practical.

The rest of this paper is organized as follows. In Section II, we review the related work. In Section III, we present the system model and the threat model, as well as introduce several necessary techniques used to design the proposed scheme. We describe the detailed implementation for the proposed *multi-authority attribute-based searchable encryption* in Section IV. Subsequently, the design of a privacy-preserving and personalized task recommendation system is presented in Section V. In Section VI, rigorous security proof and thorough security analysis are provided. In Section VII, the experimental performance evaluations are shown. Finally, Section VIII concludes this paper.

II. RELATED WORK

A. SEARCHABLE ENCRYPTION

Searchable encryption is a desirable technique that allows a server to process keyword-based search over encrypted data. Song *et al.* [8] first presented searchable encryption. Goh *et al.* [17] and Chang and Mitzenmacher [18] proposed the construction of a secure index for each encrypted file to reduce the search cost. To achieve sub-linear search complexity, Curtmola *et al.* [19] designed the encrypted inverted index construction for file collections. In that paper, the authors formally defined the information leakage functions and proved the security of the proposed searchable encryption scheme. Because the data file update was not considered, all of the above works are regarded as *static* searchable encryption. In contrast, *dynamic* search encryption supports secure and efficient data update, including data deletion and addition. Recently, several advanced *dynamic* search encryption schemes [20]–[23] have been proposed for pursuing better efficiency and security. To allow one to verify the correctness and completeness of the search results without decryption, verifiable searchable encryption construction was proposed in [24], which can effectively prevent a dishonest server from returning erroneous results. As these schemes were constructed in symmetric encryption setting, they are called the searchable symmetric encryption *SSE* scheme. The first public-key-based searchable encryption was proposed by Boneh *et al.* in [25]. Later, the searchable encryption supporting the multi-keyword conjunctive search was also explored in the public-key setting such as [26], [27], and [28].

B. ATTRIBUTE-BASED ENCRYPTION

Attribute-based encryption (ABE) is a promising public-key encryption system that allows a data encrypter to enforce fine-grained data access control over encrypted data, which is particularly suitable for data sharing in a remote untrusted environment such as a cloud storage system. Sahai and Waters [16] first presented attribute-based encryption. Subsequently, this technique was further developed into two fundamental branches: key-police attribute-based encryption (KP-ABE) [29], [30] and ciphertext-police attribute-based encryption (CP-ABE) [31]–[33]. In the KP-ABE, a ciphertext is associated with a set of attributes, while a decryption key is associated with an access policy. When the attributes in the ciphertext satisfy the access control policy in the key, the key can be used to decrypt the ciphertext. In contrast, in the CP-ABE, an encrypter encrypts a message with a specified access policy, while a decrypter's key is generated according to his/her attributes. The decrypter can decrypt the message if and only if his/her attributes satisfy the access policy in the corresponding ciphertext. The abovementioned traditional ABE systems need one central authority to distribute keys for all the users, which may incur a performance bottleneck or even a single point of failure.

To eliminate the *single* central authority, Lewko and Waters [1] proposed a multi-authority CP-ABE system. Their system does not require any central authority, where any party can become an authority to create a public key and issue private keys to different users. Obviously, this system is more suitable to be used in the secure and personalized task recommendations scenario, since there exist numerous requesters and works. In this study, we let each requester be an authority to encrypt his/her own tasks under the concerned attributes and issue keys for authorized workers according to the workers' attributes (*identities*). Thus, the application flexibility and scalability can be guaranteed better.

Similar to our idea in this paper, several novel attribute-based searchable encryption schemes were proposed recently in [34]–[39], which enable fine-grained access control and data searching over ciphertext simultaneously. However, these schemes are constructed based on the attribute-based encryption with a *single* central authority, which is not applicable to our personalized task recommendations scenario, where there exist multiple requesters and workers. To highlight the differences between our multi-authority attribute-based searchable encryption and the existing schemes above, we provide a differential comparison among these schemes, as shown in Table 1.

C. TASK RECOMMENDATION IN CROWDSOURCING

Task recommendation in crowdsourcing can guarantee that the tasks are pushed to the right workers to make the requester gain better-quality output. Chilton *et al.* [40] researched the problem of how workers obtain tasks more quickly from the MTurk platform by using the task search. Later,

TABLE 1. Comparison among schemes.

Schemes	Authority	Keyword Search	Access Structure
[1]	multiple	×	LSSS
[34]	single	√	LSSS
[35]	single	√	LSSS
[38]	single	√	access tree
[39]	single	√	access tree
Ours	multiple	√	LSSS

TABLE 2. Comparison among schemes.

Schemes	Encryption	Personalized Recommendation
[3], [14], [15]	×	√
[7], [9]–[13]	√	×
Ours	√	√

Ambati *et al.* [14] developed a task recommendation engineer that can recommend tasks to the suitable workers on the basis of the workers' skills and interests. Yuen *et al.* [3] relied on the fact that the workers' search and performance history can reflect the workers' preferences to propose a preference-based task recommendation framework. They further considered the dynamic scenarios of the workers and the task updates and proposed a task recommendation framework in [41]. Difallah *et al.* [15] proposed a novel crowdsourcing task recommendation architecture based on the worker profiles extracted from a social network, which can push each task to the right worker dynamically. Ye and Wang [42] aimed to solve the problem that dishonest workers may obtain recommendations by counterfeiting good reputations and overstating personal skills and proposed a trust-aware worker recommendation scheme for crowdsourcing environments, which can guarantee that a worker submits the correct answer with a high probability. Geiger and Schader [4] presented a systematic analysis of the personalized task recommendation in crowdsourcing as well as identified several open issues for a future research agenda. Although these works thoroughly explored the task recommendation technique for crowdsourcing, the security and privacy issues were not considered. Yang *et al.* [43] indicated that both the tasks and the workers face critical security and privacy challenges in crowdsourcing networks, when the task contents and the workers' information are exposed to the crowdsourcing platform. To preserve task confidentiality and worker privacy simultaneously, Shu and Jia [7] and Shu *et al.* [9]–[13] proposed several secure task recommendation schemes, where secure task recommendation is realized by keyword-based matching between encrypted tasks and interests. In this paper, we design a task recommendation system in crowdsourcing achieving personalized recommendation according to the worker's identities, guaranteeing the confidentiality of requester's tasks as well. To highlight the differences between our schemes and the recently related works, we provide a comparison among these schemes, as shown in Table 2.

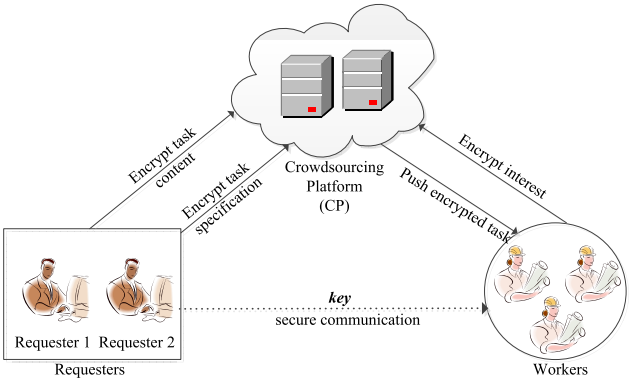


FIGURE 2. System model of privacy-preserving task recommendation in crowdsourcing.

III. PROBLEM FORMULATION

In this section, we present the system model and the threat model, and introduce several necessary techniques used to implement the proposed scheme.

A. SYSTEM MODEL

The system model includes three types of entities: multiple task requesters, multiple workers, and a crowdsourcing platform, as shown in Fig. 2.

A task requester is a task owner as well as an authority, who administers a set of attributes and issues a private key to an authorized worker according to the worker's identities. For protecting the tasks' confidentiality, before publishing them to the crowdsourcing platform, the requester encrypts the tasks and the corresponding task specifications. A specification is a task's keyword-based description, which can be seen as the search index of the task.

A worker encrypts his interests and submits the cihpertext to the crowdsourcing platform to obtain tasks.

Upon receiving a worker's encrypted interests, the crowdsourcing platform (CP) is responsible for matching encrypted tasks with encrypted interests. If the worker's interests and identities satisfy a task simultaneously, the CP pushes the task to the worker, who decrypts the task locally.

Note that there exists rigorous authentication between the requesters and the workers. A worker who has passed identity authentication can obtain a private key containing his attributes from the requesters via secure communication channels, which can be achieved by using a registration mechanism or a remote authentication protocol [44].

B. THREAT MODEL

CP is generally deployed in the remote cloud server, which is not fully trusted by the users. Therefore, we considered CP to be the "honest but curious" entity. In particular, CP honestly and correctly executes the designed protocols, but it may want to obtain as much information as possible from the requesters' tasks and workers' interests. Our security goal is to prevent the semi-trusted CP from inferring any

useful information from the published tasks and the submitted interests. In the system model, both the requesters and the workers are trusted.

C. BASIC TECHNIQUES

1) BILINEAR PAIRING MAP

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ denote two cyclic multiplicative groups of order q. Define a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ such that $e(Q, Z) \in \mathbb{G}_2$ can be efficiently calculated and $e(Q^x, Z^y) = e(Q, Z)^{xy}$ holds for any $Q, Z \in \mathbb{G}_1$ and $x, y \in \mathbb{Z}_q^*$.

2) MULTI-AUTHORITY CP-ABE

We use a set of descriptive attributes to denote a worker's identities. For example, "The worker is a *professor* with specialty *computer science*," where *professor* and *computer science* are the two attributes used to describe the worker's identities. Based on the multi-authority CP-ABE scheme in [1], we develop *multi-authority attribute-based searchable encryption* to implement privacy-preserving and personalized task recommendations in crowdsourcing. Several necessary definitions are introduced as follows.

Definition 1 (Attribute Set): Assume that there exist n requesters in the crowdsourcing platform. We define an attribute universe $S = S_1 \cup S_2 \cup \ldots \cup S_n$, *where* $S_i \subset S \setminus \emptyset$ *denotes the attribute set administrated by the* ***ith*** *requester, who takes advantage of the attribute set to encrypt tasks and generate a private key for authorized workers.*

Definition 2 (Access Structure): Let $\{P_1, P_2, \ldots, P_n\}$ *be a set of parties. A collection* $\mathbb{A} \subseteq 2^{\{P_1, P_2, \ldots, P_n\}}$ *is monotone if* $\forall B, C$: *if* $B \in \mathbb{A}$ *and* $B \subseteq C$ *then* $C \in \mathbb{A}$. *An access structure is a collection* $\mathbb{A}$ *of non-empty subsets of* $\{P_1, P_2, \ldots, P_n\}$, *i.e.,* $\mathbb{A} \subseteq 2^{\{P_1, P_2, \ldots, P_n\}} \setminus \{\emptyset\}$. *The sets in* $\mathbb{A}$ *are called the authorized sets, and the sets not in* $\mathbb{A}$ *are called the unauthorized sets.*

In an attribute-based encryption scheme, the role of the parties is represented by the attributes. Thus, the access structure $\mathbb{A}$ is an access policy, such as an access tree [29], [31], organized by different attributes from the authorized sets of attributes. The aforementioned Boolean formula used to encrypt a task can be easily expressed by an access tree, where the interior nodes are AND and OR gates and the leaf nodes are attributes.

Definition 3 (Linear Secret Sharing Scheme, LSSS): A secret sharing scheme Π *over a set of parties* $\mathcal{P}$ *is called linear if (1) the shares for each party form a vector over* $\mathbb{Z}_p$ *and (2) there exists a matrix* $\boldsymbol{A}$ *with l rows and n columns called the share generation for* π. *For all* $i = 1, \ldots, l$, *the* i^{th} *row of* $\boldsymbol{A}$ *is labeled by a party* $\rho(i)$, *where* ρ *is a function from* $\{1, \ldots, l\}$ *to* $\mathcal{P}$. *When we consider the column vector* $v = (s, r_2, \ldots, r_n)$, *where* $s \in \mathbb{Z}_p$ *the secret to be shared and* $r_2, \ldots, r_n \in \mathbb{Z}_p$ *are randomly chosen, then* $\boldsymbol{A}v$ *is the vector of l shares of the secret s according to* π. *The share* $(Av)_i$ *belongs to party* $\rho(i)$.

In [1], LSSS is used to represent an access structure, where π is an LSSS for the access structure $\mathbb{A}$. Suppose that $S \in \mathbb{A}$ is any authorized set and we define $I = \{i : \rho(i) \in S\} \subset \{1, 2, \ldots, l\}$. We can find constants $\{w_i \in \mathbb{Z}_p\}_{i \in I}$ in the time polynomial in the size of the share-generation matrix A such that $\sum_{i \in I} w_i \lambda_i = s$ holds, for any valid shares $\{\lambda_i\}$ of a secret s according to π. For unauthorized sets, no such constants $\{w_i\}$ exist.

3) DECISIONAL DIFFIE-HELLMAN (DDH) PROBLEM AND ASSUMPTION

DDH Problem: Let g represent a generator of the group $\mathbb{G}$ with order q. There are three random elements a, b, and c in $\mathbb{Z}_q^*$. Given (g, g^a, g^b), the problem is to distinguish the valid element g^{ab} from the random element g^c. A PPT algorithm $\mathcal{A}$ has an advantage $Adv_{\mathcal{A}}^{\mathrm{DDH}}$ in solving the DDH problem if

$$Adv_{\mathcal{A}}^{\mathrm{DDH}} \leq |Pr[\mathcal{A}(g^a, g^b, g^{ab}) = 1] - Pr[\mathcal{A}(g^a, g^b, g^c) = 1]|$$

DDH assumption: For any probabilistic polynomial time algorithm $\mathcal{A}$, $Adv_{\mathcal{A}}^{\mathrm{DDH}}$ is negligible.

IV. MULTI-AUTHORITY ATTRIBUTE-BASED SEARCHABLE ENCRYPTION

In this section, we discuss the development of a multi-authority attribute-based searchable encryption scheme, which is the core block to implement the privacy-preserving and personalized task recommendation in crowdsourcing. We first present the formal definition and then propose the concrete construction. Finally, we provide the correctness analysis of the scheme.

A. DEFINITION

*Definition 4: The multi-authority attribute-based searchable encryption (****MASE****) consists of the following six polynomial-time algorithms:*

- ***MASE.Init****(1^k). The algorithm takes a security parameter k as the input and outputs a global parameter GP.*
- ***MASE.SetUp****(GP). The algorithm is invoked by each authority with GP as the input and generates its own private/public key pair, PK, SK.*
- ***MASE.KeywordEnc****(w, (A, ρ), PK, GP). The secure index generation algorithm takes an index keyword w, an LSSS matrix (A, ρ), the public key PK of the authority, and the global parameter GP as the inputs and outputs the ciphertext of the index keyword,* ind_w.
- ***MASE.KenGen****(SU, GP, a, SK). The algorithm is invoked by an authority administering attribute a with the private key SK and generates an attribute key,* $SK_{SU,a}$, *associated with attribute a for SU, where parameter SU denotes a search user with attribute a, GP is the global parameter.*
- ***MASE.TrapdoorGen****(*$\{SK_{SU,a}\}$*, q). An authorized search user SU with a set of attributes {a} takes the corresponding attribute key set* $\{SK_{SU,a}\}$ *and a query keyword q as the inputs, and the algorithm generates the ciphertext of the query keyword q,* Trp_q.
- ***MASE.Match****(*ind_w, Trp_q*). The algorithm takes the encrypted index keyword* ind_w *and encrypted query*

MASE.Init(1^k). Input a security parameter k, choose two cyclic multiplication groups $\mathbb{G}_1$ and $\mathbb{G}_2$ with the same prime order p, and define a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ between the two groups and two hash functions $H_1 : \{0,1\}^* \to \mathbb{Z}_p^*$ and $H_2 : \{0,1\}^* \to \mathbb{G}_1$ hashing a string of arbitrary length to a valid element in $\mathbb{Z}_p^*$ and $\mathbb{G}_1$, respectively. Let g be a generator of group $\mathbb{G}_1$; then, the system global parameter is GP=$\{\mathbb{G}_1, \mathbb{G}_2, H_1, H_2, e, g\}$.

MASE.SetUp(GP). Let S_{AU} be a set of attributes administrated by an authority AU. For each attribute $a \in S_{\text{AU}}$, AU chooses two element $x_a, y_a \in \mathbb{Z}_p^*$ uniformly at random and computes g^{x_a}, g^{y_a}. AU publishes the public key PK=$\{\forall a \in S_{\text{AU}} : (g^{x_a}, g^{y_a})\}$, the private key SK=$\{\forall a \in S_{\text{AU}} : (x_a, y_a)\}$ is kept secret.

MASE.KeywordEnc(w, (A, ρ), PK, GP). The algorithm encrypts an index keyword w under the public key PK of an authority AU, global parameter GP, and an $n \times l$ matrix A with ρ mapping its rows to attributes. It randomly chooses an element $s \in \mathbb{Z}_p^*$ and a vector $v = (s, b_2, ..., b_l) \in \mathbb{Z}_p^l$, where $b_2, ...b_l$ are used to share the secret s. Let A_i denote the i-th row of matrix A. For $i = 1$ to n, it computes the inner product $\lambda_i = A_i \cdot v$ as well as chooses a random vector $h \in \mathbb{Z}_p^l$ with 0 as its first entry and computes $h_i = A_i \cdot h$. For each row A_i of A, it chooses a random element r_i and encrypts the index keyword w as

$$\text{ind}_w = \Big(\mathcal{I} = e(g,g)^s, \text{For } i = 1 \text{ to } n : \mathcal{I}_{1,i} = e(g,g)^{\lambda_i}, \mathcal{I}_{2,i} = g^{x_{\rho(i)} r_i} H_1(w),$$
$$\mathcal{I}_{3,i} = g^{r_i}, \mathcal{I}_{4,i} = g^{y_{\rho(i)} r_i} H_1(w), \mathcal{I}_{5,i} = g^{h_i}\Big)$$

MASE.KenGen(SU, GP, a, SK). We use ID_{SU} to denote the search user SU. By invoking the algorithm, AU, who administers the attribute a, generates a key $\text{SK}_{\text{SU},a} : (K_{1,a} = g^{x_a}, K_{2,a} = H_2(ID_{\text{SU}})^{y_a})$ by using his/her private key SK, and issues it to SU via secure communication channels. Let S_{SU} denote a set of attributes of search user SU; then, SU's attribute key can be denoted as follows:

$$\text{SK}_{\text{SU}} = \Big(\forall a \in S_{\text{SU}}, \text{SK}_{\text{SU},a} : (K_{1,a} = g^{x_a}, K_{2,a} = H_2(ID_{\text{SU}})^{y_a})\Big)$$

MASE.TrapdoorGen(SK_{SU}, q). The authorized search user US (denoted by ID_{SU}) first chooses a value t from $\mathbb{Z}_q^*$ at random uniformly and computes g^t. Then, US uses his/her key SK_{SU} and the random value t to encrypt query keyword q to generate the query trapdoor as follows.

$$\begin{aligned}\text{Trp}_q &= \Big(T = g^t, T' = H_2(ID_{\text{SU}}), T'' = H_2(ID_{\text{SU}})^t, \forall a \in S_{\text{SU}}, T_1 = (K_{1,a})^{tH_1(q)}, T_2 = (K_{2,a})^{tH_1(q)}\Big)\\ &= \Big(T = g^t, T' = H_2(ID_{\text{SU}}), T'' = H_2(ID_{\text{SU}})^t, \forall a \in S_{\text{SU}}, T_1 = g^{t x_a H_1(q)}, T_2 = H_2(ID_{\text{SU}})^{t y_a H_1(q)}\Big)\end{aligned}$$

MASE.Match(ind_w, Trp_q). The algorithm performs matching between the encrypted index keyword and the trapdoor. If the following two conditions, (1) the trapdoor Trp_q submitted by SU satisfies the LSSS matrix (A, ρ) embedded into ind_w and (2) $w = q$, hold simultaneously, the algorithm returns 1; otherwise, it returns 0. If condition (1) holds, the algorithm can find a subset of rows A_i of A such that $(1, 0, ..., 0)$ is in the span of these rows and further specifies a constant $c_i \in \mathbb{Z}_p$ such that $\sum_i c_i A_i = (1, 0, ..., 0)$. Then, the algorithm decides whether the following equation is true.

$$\prod_i \left(\frac{\text{ind}_w.\mathcal{I}_{1,i} \cdot e(\text{ind}_w.\mathcal{I}_{2,i}, \text{Trp}_q.T) \cdot e(\text{Trp}_q.T'', \text{ind}_w.\mathcal{I}_{4,i}) \cdot e(\text{Trp}_q.T', \text{ind}_w.\mathcal{I}_{5,i})}{e(\text{Trp}_q.T_1 \cdot \text{Trp}_q.T_2, \text{ind}_w.\mathcal{I}_{3,i})}\right)^{c_i} = \text{ind}_w.\mathcal{I}$$

If the above equation holds, which implies that $w = q$, the algorithm returns 1.

FIGURE 3. Multi-authority attribute-based searchable encryption construction.

keyword Trp_q as the inputs, and outputs 1 if $w = q$ and the trapdoor Trp_q satisfies the LSSS access matrix (A, ρ) embedded into the index ind_w; else, it outputs 0.

B. CONSTRUCTION

In this subsection, we present the concrete construction for MASE, as shown in Fig. 3.

C. CORRECTNESS

We conduct a correctness analysis for the proposed **MASE** scheme. In fact, **MASE** achieves authorized keyword search over encrypted data upon the introduction of attribute-based encryption. Given an index keyword w and a query keyword q, a successful search needs to satisfy two conditions: one is that the query trapdoor Trp_q associated with a set of attributes must satisfy the LSSS matrix embedded into ind_w, and the other condition is $w = q$. If the two conditions hold simultaneously, the algorithm **MASE.Match** can return 1 correctly. The correctness analysis is shown in Fig. 4, where trapdoor Trp_q is generated by the authorized search user SU denoted by ID_{SU}.

V. SECURE AND PERSONALIZED TASK RECOMMENDATION SYSTEM IN CROWDSOURCING

A. CONSTRUCTION

In this section, we discuss the use of the proposed MASE to implement a *privacy-preserving and personalized task recommendation system* (3PTRS) in crowdsourcing. The system

Without loss of generality, let ind_w be the encryption of an index keyword w and Trp_q submitted by search user SU denote the query trapdoor of a query keyword q, which is the output of algorithm **MASE.KeywordEnc** $(w, (A, \rho)$, PK, GP) and algorithm **MASE.TrapdoorGen**(SK_{SU}, q), respectively. According to the matching algorithm **MASE.Match** on input ind_w and Trp_q, if $w = q$, we have

$$\begin{aligned}
&\prod_i \Big(\frac{\text{ind}_w.\mathcal{I}_{1,i} \cdot e(\text{Trp}_q.T, \text{ind}_w.\mathcal{I}_{2,i}) \cdot e(\text{Trp}_q.T'', \text{ind}_w.\mathcal{I}_{4,i}) \cdot e(\text{Trp}_q.T', \text{ind}_w.\mathcal{I}_{5,i})}{e(\text{Trp}_q.T_1 \cdot \text{Trp}_q.T_2, \text{ind}_w.\mathcal{I}_{3,i})}\Big)^{c_i} \\
&= \prod_i \Big(\frac{e(g,g)^{\lambda_i} \cdot e(g^t, g^{x_{\rho(i)} r_i H_1(w)}) \cdot e(H_2(ID_{\text{SU}})^t, g^{y_{\rho(i)} r_i H_1(w)}) \cdot e(H_2(ID_{\text{SU}}), g^{h_i})}{e(g^{t x_{\rho(i)} H_1(q)} \cdot H_2(ID_{\text{SU}})^{t y_{\rho(i)} H_1(q)}, g^{r_i})}\Big)^{c_i} \\
&= \prod_i \Big(\frac{e(g,g)^{\lambda_i} \cdot e(g,g)^{t x_{\rho(i)} r_i H_1(w)} \cdot e(H_2(ID_{\text{SU}}), g)^{t y_{\rho(i)} r_i H_1(w)} \cdot e(H_2(ID_{\text{SU}}), g^{h_i})}{e(g,g)^{t x_{\rho(i)} r_i H_1(q)} \cdot e(H_2(ID_{\text{SU}}), g)^{t y_{\rho(i)} r_i H_1(q)}}\Big)^{c_i} \\
&= \prod_i \Big(e(g,g)^{\lambda_i} \cdot e(H_2(ID_{\text{SU}}), g)^{h_i}\Big)^{c_i} \\
&= \prod_i \Big(e(g,g)^{A_i \cdot v \cdot c_i} \cdot e(H_2(ID_{\text{SU}}), g)^{A_i \cdot h \cdot c_i}\Big)
\end{aligned} \tag{1}$$

If the query trapdoor Trp_q satisfies the LSSS matrix (A, ρ) embedded into ind_w, we can find a group of constants $c_i (1 \le i \le n)$ such that $\sum_i^n c_i A_i = (1, 0, ..., 0)$, where A_i is ith row of matrix A and n is the total number of rows. According to equation (1), we can further deduce

$$\begin{aligned}
&\prod_i \Big(e(g,g)^{A_i \cdot v \cdot c_i} \cdot e(H_2(ID_{\text{SU}}), g)^{A_i \cdot h \cdot c_i}\Big) \\
&= e(g,g)^{\sum_i^n A_i \cdot v \cdot c_i} \cdot e(H_2(ID_{\text{SU}}), g)^{\sum_i^n A_i \cdot h \cdot c_i} \\
&= e(g,g)^{(1,0,...,0) \cdot (s, b_1, ..., b_l)} \cdot e(H_2(ID_{\text{SU}}), g)^{(1,0,...,0) \cdot (0, r_1, ..., r_l)} \\
&= e(g,g)^s \cdot e(H_2(ID_{\text{SU}}), g)^0 \\
&= e(g,g)^s \\
&= \text{ind}_w.\mathcal{I}
\end{aligned} \tag{2}$$

Therefore, according to equations (1) and (2), we can ensure that if the query trapdoor Trp_q associated with a set of attributes satisfies the LSSS matrix embedded into ind_w and $w = q$, algorithm **MASE.Match** can return 1 correctly.

FIGURE 4. **Correctness analysis of MASE.**

can ensure that an encrypted task is recommended to the most suitable worker according to the worker's interests and identities (*attributes*), where the information of both the requester and the worker is well protected by encryption against the semi-trusted crowdsourcing platform.

Next, we describe the 3PTRS system specification from the module level, as shown in Fig. 5, where each module of 3PTRS invokes one or more algorithms of MASE.

B. DISCUSSION

1) IDENTITY-HIDDEN TASK RECOMMENDATION

Thus far, in the **Task Recommendation** module of 3PTRS, an important problem has not been solved. An observant reader will realize that it is difficult for CP to find a subset of rows A_i from an LSSS matrix encrypting a task such that $(1, 0, \ldots, 0)$ is in the span of these rows based on the worker's submitted trapdoor even if the worker's attributes satisfy the LSSS access matrix, as the worker's attributes in the trapdoor are invisible in terms of CP. A straightforward approach to solve this problem is that the worker submits a trapdoor along with his/her attributes to CP together. However, the process also means that the worker's identity information is exposed to CP, which compromises the worker's privacy. To allow CP to correctly perform algorithm **MASE.Match** as well as protect the worker's privacy, we propose to compute each attribute's keyed-hash message authentication code (HMAC) and map HMAC of an attribute (instead of the attribute itself) to a row of the LSSS matrix by using map function ρ. Correspondingly, the worker computes HMACs of his/her attributes and submits a trapdoor and HMACs together to CP. Thus, CP can easily determine whether a worker's attributes satisfy the LSSS matrix encrypting a task or not, while maintaining the confidentiality of the worker's attributes. Familiar HMAC implementations include HMAC-MD5 and HMAC-SHA1, which are semantically secure and computationally efficient [46]. A concrete example is shown in Fig. 6. In addition, an access formula can be efficiently converted to an LSSS matrix by using the algorithm proposed in [1].

We can observe that, in the **Task Recommendation** module, besides the matching between the encrypted interests and

System Initiation. The system initiation module selects a large security parameter l and runs algorithm **MASE.Init**(l) to generate public parameters GP=$\{\mathbb{G}_1, \mathbb{G}_2, H_1, H_2, e, g, q\}$.

Task Encryption and Publishing. The functionality of this module is to achieve task encryption and publishing. For a requester R and his a task T, to achieve task recommendation, R extracts descriptive keywords from T as the task specification. For the sake of simplicity, we use one keyword Q to denote T's task specification. In this phase, R first runs algorithm **MASE.SetUp**(GP) to output his/her public/private pair $\mathrm{PK}_R, \mathrm{SK}_R$. Then, R specifies a Boolean formula containing n attributes, converts the Boolean formula into an $n \times l$ LSSS matrix (A, ρ), and encrypts task specification Q by invoking algorithm **MASE.KeywordEnc**$(Q, (A, \rho), \mathrm{PK}_R, \mathrm{GP})$ to obtain the ciphertext ind_Q. Further, R chooses a random seed $\sigma \in \mathbb{G}_2$ to generate an AES encryption key κ by invoking **AES.KeyGen**(σ) and encrypts task T as C_T by running **AES.Enc**(κ, T). Finally, R encrypts the random seed σ with the LSSS matrix (A, ρ) under the public key PK_R as follows:

$$C_\sigma = \Big((A, \rho), C = \sigma e(g,g)^t, \text{For } i = 1 \text{ to } n : C_{1,i} = e(g,g)^{\lambda_i} \cdot e(g,g)^{x_{\rho(i)} r_i}, C_{2,i} = g^{r_i}, C_{3,i} = g^{y_{\rho(i)} r_i} \cdot g^{h_i}\Big),$$

where t is a randomly chosen element in $\mathbb{Z}_q^*$, $\lambda_i = A_i \cdot v$, A_i is row i of A and $v \in \mathbb{Z}_q^l$ is a vector with t as its first entry, r_i is a randomly chosen element for each row A_i, $h_i = A_i \cdot h$, and $h \in \mathbb{Z}_q^l$ is a random vector with 0 as its first entry.
Finally, R sends ciphertext $\mathbb{C}(T) = (\mathrm{ind}_Q, C_T, C_\sigma)$ to CP.

Worker Enrollment. This module is responsible for new user registration. A new worker enrolls into the system to obtain an authorized key on the basis of the worker's attributes, using which the worker encrypts his/her interests. Assume that a worker W with a set of attributes S_W wants to join the system, he/she first certifies the authenticity of his/her identities by using a registration mechanism [45] or remote authentication [44]. Then, for each attribute $a \in S_W$ a requester R administers, R runs the algorithm **MASE.KenGen**$(W, \mathrm{GP}, a, \mathrm{SK}_R)$ to generate a key $\mathrm{SK}_{W,a}$ and sends it to W via secure communication channels. Finally, the requester-authorized attribute key for the worker is denoted by $\mathrm{SK}_W = (\forall a \in S_W, \mathrm{SK}_{W,a})$, which is used to encrypt his/her interests by the worker.

Interest Encryption. When a worker W joins the system successfully, he/she runs this module to encrypt his/her interests. For the sake of simplicity, we use q to denote W's one interest, W invokes algorithm **MASE.TrapdoorGen**(SK_W, q) to obtain the encrypted interest Trp_q. Obviously, if the worker has n interests, he/she can run n times **MASE.TrapdoorGen** to obtain the corresponding encryption version.

Task Recommendation. This module is deployed in CP and is responsible for recommending an encrypted task to the most suitable worker according to the worker's submitted interests. Given an encrypted task T with task specification Q and a worker W with encrypted interest Trp_q, the module runs algorithm **MASE.Match**$(\mathbb{C}(T).\mathrm{ind}_Q, \mathrm{Trp}_q)$. If the algorithm returns 1, then $\mathbb{C}(T)$ is returned to W.

Task Decryption. When a worker W with key $\mathrm{SK}_W = (\forall a \in S_W, K_1 = g^{x_a}, K_2 = H_2(ID_W)^{y_a})$ receives an encrypted task $\mathbb{C}(T)$, he takes the following steps to decrypt the ciphertext to obtain T. First, the worker W finds a subset of rows A_i of $\mathbb{C}(T).C_\sigma.A$ on the basis of his attribute set S_W such that $(1, 0, ..., 0)$ is in the span of these rows, and computes the following for each such i:

$$\frac{C_\sigma.C_{1,i} \cdot e(H_2(ID_W), C_\sigma.C_{3,i})}{e(g^{x_{\rho(i)}} \cdot H_2(ID_W)^{y_{\rho(i)}}, C_\sigma.C_{2,i})} = e(g,g)^{\lambda_i} e(H_2(ID_W), g)^{h_i}$$

The worker further chooses a constant c_i such that $\sum_i c_i A_i = (1, 0, ..., 0)$ and computes the following:

$$\prod_i \Big(e(g,g)^{\lambda_i} e(H_2(ID_W), g)^{h_i}\Big)^{c_i} = e(g,g)^t$$

where $\lambda_i = A_i \cdot v$, $h_i = A_i \cdot h$, $v \cdot (1, 0, ..., 0) = t$, and $h \cdot (1, 0, ..., 0) = 0$. Second, the worker W computes the following:

$$\frac{\mathbb{C}(T).C_\sigma.C}{e(g,g)^t} = \frac{\sigma \cdot e(g,g)^t}{e(g,g)^t} = \sigma$$

Finally, W runs algorithm **AES.KeyGen**(σ) to generate the task encryption key κ and decrypts the task by running **AES.Dec**$(\kappa, \mathbb{C}(T).C_T)$. Note that the worker W can certainly find a subset of rows A_i of $\mathbb{C}(T).C_\sigma.A$ such that $(1, 0, ..., 0)$ is in the span of these rows to recover the secret share t in the LSSS structure as long as the task T is recommended to him/her, as the identical access matrix (A, ρ) is used to encrypt task specification Q and random seed σ. Meanwhile, the identical key SK_W is used to encrypt W's interests and decrypt the random seed σ.

FIGURE 5. The privacy-preserving and personalized task recommendation system.

the task specifications, the matching between a task's access policy and a worker's identities is implicitly performed. Thus, only if a worker's interests and identities match a task specification simultaneously, the encrypted task could be recommended to the worker, and the goal of secure and personalized recommendation is achieved.

2) WORKER REVOCATION

Efficiently and flexibly revoking a worker's task request capability is an important property in the dynamic crowdsourcing environment. To achieve this, we propose to use symmetric encryption and group key idea. Let $\mathsf{SE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be the traditional symmetric encryption scheme such as AES. Specifically, the crowdsourcing platform generates a key k by invoking the key generation algorithm $\mathsf{SE.Gen}$, which is shared between the crowdsourcing platform and workers securely. A work with attribute $\{a, b, d\}$, as mentioned in Fig. 6, generates the trapdoor component Trap by running the encryption algorithm $\mathsf{SE.Enc}(k, \text{HMAC}(a)\|\text{HMAC}(b)\|\text{HMAC}(d))$, where $\|$ denote the concatenation of two strings. Obviously, before performing attribute match, the crowdsourcing platform can recover HMAC(a), HMAC(b), and HMAC(d) via the decryption algorithm $\mathsf{SE.Dec}(k, \text{Trap})$. When an authorized worker is revoked from the system, the crowdsourcing platform only needs to update the group key k to k' and distributes it to unrevoked workers via secure channels. As a result, the revoked worker cannot generate valid trapdoor component without the updated key k'. To guarantee the user experience, when a worker is revoked, the crowdsourcing platform will send a notification to the worker.

VI. SECURITY ANALYSIS

The security goal of the proposed scheme is to preserve the privacy of the requester's tasks and the worker's interests against the ''honest but curious'' crowdsourcing platform. In particular, given a task T with task specification Q and a worker's interest q, CP is prohibited from gaining any useful information from their encrypted versions. This goal can be easily achieved for task T because of the semantically secure symmetric encryption AES. Therefore, we focus on analyzing the security of the task specifications and the worker's interests.

To provide formal security proof, we first define the Adaptively Chosen-Keyword Attack Model by the following game between an challenger $\mathcal{B}$ and a polynomial-time adversary $\mathcal{A}$.

Setup. The Challenger $\mathcal{B}$ sets up system global parameter **GP** by running algorithm **Init** and sends **GP** to the adversary $\mathcal{A}$.

Phase 1.

- $\mathcal{A}$ requests the trapdoor Trp_{q_i} of any keyword $\{q_i\}_{1\le i\le n}$ by adaptively query oracle $\mathcal{O}_{\text{KenGen}}$ and $\mathcal{O}_{\text{TrapdoorGen}}$ oracle for polynomially many times with the attribute sets $S_1, \ldots, S_n$.
- $\mathcal{A}$ requests the ciphertext ind_{w_j} of any keyword $\{w_j\}_{1\le j\le m}$ by adaptively query oracle $\mathcal{O}_{\text{KeywordEnc}}$ for polynomially many times with the LSSS matrices $A_1, \ldots, A_m$.
- $\mathcal{A}$ queries oracle $\mathcal{O}_{\text{Match}}(\text{ind}_{w_j}, \text{Trp}_{q_i})_{1\le j\le m, 1\le i\le n}$ for polynomially many times. If S_i satisfies A_j and $q_i = w_j$, the oracle outputs 1; otherwise returns 0.

Challenge. $\mathcal{A}$ defines a challenge LSSS matrix A^* such that none of the attribute sets $S_1, \ldots, S_n$ from Phase 1 satisfy A^*. $\mathcal{A}$ sends two keywords w_0 and w_1 along with A^* to $\mathcal{B}$. $\mathcal{B}$ flips a random binary coin $b \in \{0, 1\}$ and encrypts w_b with A^* as ind_{w_b}, which is sent to $\mathcal{A}$.

Phase 2. $\mathcal{A}$ repeats **Phase 1.** for any keyword $\{q_i\}_{n+1\le i\le o}$ with the attribute sets $S_{n+1}, \ldots, S_o$ and any keyword $\{w_j\}_{m+1\le j\le o}$ with the LSSS matrices $A_{m+1}, \ldots, A_o$. The only restriction is that $A^* \notin \{A_k\}_{1\le k\le o}$ and if the attribute set S_j for keyword q_j satisfies the challenge LSSS matrix A^*, then $q_j \ne w_0, w_1$.

Guess. $\mathcal{A}$ outputs a guess b' of b.

The advantage that a probabilistic polynomial time adversary $\mathcal{A}$ wins the game is defined as $Adv = \mathbf{Pr}[b = b'] - \frac{1}{2}$.

Definition 5: The multi-authority attribute-based searchable encryption we propose in this work is semantically secure against an adaptively chosen-keyword attack if the advantage Adv in winning the above game for any probabilistic polynomial time adversary is negligible.

A. SECURITY PROOF

Theorem 1: Our proposed multi-authority attribute-based searchable encryption (MASE) is semantically secure against the adaptively chosen-keyword attack in the generic bilinear group model.

Proof: Intuitively, the adversary is able to correctly output the guess b' of b for the challenge keyword w_b with probability 1 by querying the oracle $\mathcal{O}_{\text{Match}}(\text{ind}_{w_b}, \text{Trp}_{w_0}) = 1$ or $\mathcal{O}_{\text{Match}}(\text{ind}_{w_b}, \text{Trp}_{w_1}) = 1$. Next, we prove that this is an impossible event in the adaptively chose-keyword attack game defined above based on the generic bilinear group model. I.e., the adversary $\mathcal{A}$ always only queries the results $\mathcal{O}_{\text{Match}}(\text{ind}_{w_b}, \text{Trp}_{w_0}) = 0$ and $\mathcal{O}_{\text{Match}}(\text{ind}_{w_b}, \text{Trp}_{w_1}) = 0$ for challenge keywords w_0 and w_1 chosen by him. As a result, $\mathcal{A}$ can only guess the correct b' such that $b' = b$ with probability at most $\frac{1}{2}$.

Setup. The challenger $\mathcal{B}$ runs algorithms **Init** and **SetUp**, and sends public parameter $\text{GP} = \{\mathbb{G}_1, \mathbb{G}_2, H_1, H_2, e, g, q\}$.

Phase 1. Initialize two lists $\mathcal{K} = \varnothing$ and $\widehat{e} = \varnothing$, in $\mathcal{K}$ each element is a tuple with two random values from $\mathbb{Z}_q^*$ and in $\widehat{e}$ each element is a keyword. When $\mathcal{A}$ queries the ith trapdoor Trp_{q_i} with the corresponding attribute set S_i, two random values t and $h_{ID_{\mathcal{A}}}$ are chosen from $\mathbb{Z}_q^*$ and $\mathcal{B}$ computes $g^{h_{ID_{\mathcal{A}}}}$ as the response to $H_2(ID_{\mathcal{A}})$. For each attribute $a \in S_i$, $\mathcal{B}$ computes $T_1 = g^{t x_a H_1(q_i)}$ and $T_2 = g^{h_{ID_{\mathcal{A}}} t y_a H_1(q_i)}$, where (x_a, y_a) is taken from $\mathcal{K}$ if (x_a, y_a) is in $\mathcal{K}$; otherwise, x_a and y_a are chosen from $\mathbb{Z}_q^*$ at random uniformly and $\mathcal{B}$ adds the new tuple (x_a, y_a) to $\mathcal{K}$, i.e., $\mathcal{K} = \mathcal{K} \cup (x_a, y_a)$.

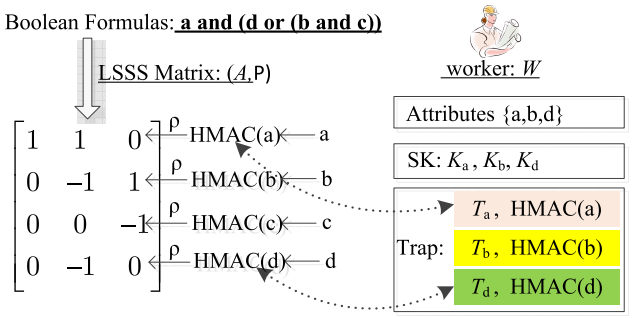


FIGURE 6. Concrete example of the privacy-preserving attribute match.

Then $\mathcal{B}$ returns the following tuple as the response to Trp_{q_i}: $(T = g^t, T' = g^{h_{ID_{\mathcal{A}}}}, T'' = g^{h_{ID_{\mathcal{A}}}t}, \forall a \in S_i, T_1 = g^{tx_aH_1(q)}, T_2 = g^{h_{ID_{\mathcal{A}}}ty_aH_1(q_i)})$. Obviously, from the adversary $\mathcal{A}$' perspective, Trp_{q_i} is a legal trapdoor construction with respect to the attribute set S_i and the keyword q_i. $\mathcal{B}$ adds the queried keyword q_i to $\widehat{e}$, i.e., $\widehat{e} = \widehat{e} \cup q_i$.

When $\mathcal{A}$ queries the jth ciphertext ind_{w_j} with respect to the keyword w_j, if $w_j \in \widehat{e}$ corresponding to q_i, $\mathcal{B}$ defines an $n \times l$ LSSS matrix A_j with ρ mapping its each row k to a corresponding attribute (denoted by a, i.e., $\rho(k) = a$) such that S_i satisfies A_j, $\mathcal{B}$ returns the following tuple as the response to ind_{w_j}. $(\mathcal{I} = e(g,g)^s$, For $i = 1$ to $n : \mathcal{I}_{1,i} = e(g,g)^{\lambda_i}, \mathcal{I}_{2,i} = g^{x_a r_i H_1(w)}, \mathcal{I}_{3,i} = g^{r_i}, \mathcal{I}_{4,i} = g^{y_a r_i H_1(w)}, \mathcal{I}_{5,i} = g^{h_i})$, where (x_a, y_a) is taken from $\mathcal{K}$ if (x_a, y_a) is in $\mathcal{K}$; otherwise, x_a and y_a are chosen from $\mathbb{Z}_q^*$ at random uniformly and $\mathcal{B}$ adds the new tuple (x_a, y_a) to $\mathcal{K}$, i.e., $\mathcal{K} = \mathcal{K} \cup (x_a, y_a)$.

Obviously, given the above specified ciphertexts ind_{w_j} and Trp_{q_i} for certain i, j, $\mathcal{A}$ can determine the key information that $w_j = q_i$ by letting oracle $\mathcal{O}_{\mathrm{Match}}(\mathrm{ind}_{w_j}, \mathrm{Trp}_{q_i})$ output 1.

Challenge $\mathcal{A}$ defines a challenge LSSS matrix A^* with ρ mapping its each row i to a corresponding attribute (denoted by a, i.e., $\rho(i) = a$) such that none of the attribute sets $S_1, \ldots, S_n$ from Phase 1 satisfy A^*. $\mathcal{A}$ sends two keywords w_0 and w_1 along with A^* to $\mathcal{B}$. $\mathcal{B}$ flips a random binary coin $b \in \{0, 1\}$ and encrypts w_b with A^* as:

$$\mathrm{ind}_{w_b} = (\mathcal{I} = e(g,g)^s, \text{For } i = 1 \text{ to } n : \mathcal{I}_{1,i} = e(g,g)^{\lambda_i}, \mathcal{I}_{2,i} = g^{x_a r_i H_1(w)}, \mathcal{I}_{3,i} = g^{r_i}, \mathcal{I}_{4,i} = g^{y_a r_i H_1(w)}, \mathcal{I}_{5,i} = g^{h_i}),$$

where $\{r_i\}_{1 \le i \le n}$, s are randomly chosen from $\mathbb{Z}_q^*$, and $\{(x_a, y_a) | \rho(i) = a\}_{1 \le i \le n}$ is taken from $\mathcal{K}$ if (x_a, y_a) is in $\mathcal{K}$; otherwise, x_a and y_a are chosen from $\mathbb{Z}_q^*$ at random uniformly.

Phase 2. $\mathcal{A}$ repeats Phase 1. The only restriction is that the challenge LSSS matrix A^* has never been queried and if certain queried attribute set S for certain keyword q satisfies A^*, then $q \neq w_0, w_1$.

Guess. In this phase, $\mathcal{A}$ decides $b = 0$ or $b = 1$ by letting $\mathcal{O}_{\mathrm{Match}}(\mathrm{ind}_{w_b}, \mathrm{Trp}_q)$ output 1. There are the following two cases in this query.

(1) $q = w_0$ or $q = w_1$. Without loss of generality, assume that $b = 0$ and $q = w_0$. According to our match algorithm, we have

$$\prod_i \left(\frac{\mathcal{I}_{1,i} \cdot e(T, \mathcal{I}_{2,i}) \cdot e(T'', \mathcal{I}_{4,i}) \cdot e(T', \mathcal{I}_{5,i})}{e(T_1 \cdot T_2, \mathcal{I}_{3,i})} \right)^{c_i}$$
$$\xrightarrow{w_0 = q}$$
$$= \prod_i \left(e(g,g)^{\lambda_i} \cdot e(H_2(ID_{\mathrm{SU}}), g)^{h_i} \right)^{c_i}$$
$$\neq e(g,g)^s$$
$$\neq \mathrm{ind}_{w_0}.\mathcal{I},$$

which leads to $\mathcal{O}_{\mathrm{Match}}(\mathrm{ind}_{w_0}, \mathrm{Trp}_{q=w_0}) = 0$. This is because that the attribute set S in Trp_q does not satisfy the challenge LSSS matrix A^* for the ciphertext ind_{w_0} according to the adaptively chosen-keyword game. Therefore, the algorithm cannot find a subset of rows A_i^* of A^* such that $(1, 0, \ldots, 0)$ is in the span of these rows to recover the secret share s. The essential security guarantee is that if the attribute set does not satisfy the LSSS matrix, no polynomial-time adversary can recover the secret s with non-negligible advantage, please refer to [1] to obtain formal security proof in the generic bilinear group model.

(2) $q \neq w_0, w_1$. Without loss of generality, assume that $b = 0$. According to our match algorithm, we have

$$\prod_i \left(\frac{\mathcal{I}_{1,i} \cdot e(T, \mathcal{I}_{2,i}) \cdot e(T'', \mathcal{I}_{4,i}) \cdot e(T', \mathcal{I}_{5,i})}{e(T_1 \cdot T_2, \mathcal{I}_{3,i})} \right)^{c_i}$$
$$\xrightarrow{w_0 \neq q}$$
$$\neq \prod_i \left(e(g,g)^{\lambda_i} \cdot e(H_2(ID_{\mathrm{SU}}), g)^{h_i} \right)^{c_i}$$
$$\neq e(g,g)^s$$
$$\neq \mathrm{ind}_{w_0}.\mathcal{I},$$

obviously, $\mathcal{O}_{\mathrm{Match}}(\mathrm{ind}_{w_0}, \mathrm{Trp}_q) = 0$.

As a result, $\mathcal{A}$ can only guess the correct b' such that $b' = b$ with probability at most $\frac{1}{2}$. □

On the other hand, the CP can obtain the plaintext Q and q by directly breaking the encryption block $\mathrm{ind}_Q.\mathcal{I}_{2,i} = g^{x_{\rho(i)} r_i H_1(Q)}$, $\mathrm{ind}_Q.\mathcal{I}_{4,i} = g^{y_{\rho(i)} r_i H_1(Q)}$, $\mathrm{Trp}_q.T_1 = g^{tx_a H_1(q)}$, and $\mathrm{Trp}_q.T_2 = H_2(ID_{SU})^{ty_a H_1(q)}$, where $x_{\rho(i)}(x_a)$, $y_{\rho(i)}(y_a)$ are the secret keys, r_i and t are random values, H_2 hashes an element to $\mathbb{G}_1 (g \in \mathbb{G}_1)$. Obviously, all of them are the same encryption constructions. For the ease of proof, we uniformly define an encryption function as follows:

$E(m) = g^{xrH_1(m)}$, $\mathrm{pk} = g^x$, $\mathrm{sk} = x$, r is a randomly chosen element for the plaintext message m.

Theorem 2: The encryption E is semantically secure against chosen plaintext attack [46] if the DDH assumption holds.

Proof: Suppose that there exists a probabilistic polynomial time adversary $\mathcal{A}$ that has a non-negligible advantage ϵ to break E; then, we can construct a simulator $\mathcal{B}$ that can solve the DDH problem with a non-negligible advantage $\frac{\epsilon}{2}$.

The challenger $\mathcal{C}$ first flips a binary coin μ. If $\mu = 0$, $\mathcal{C}$ sets tuple $t_0 : (g, A = g^a, B = g^b, C = g^{ab})$; if $\mu = 1$, he/she sets tuple $t_1 : (g, A = g^a, B = g^b, C = g^c)$, where $a, b, andc$ are

chosen from $\mathbb{Z}_q^*$ with random uniformity. Tuple t_μ is sent to simulator $\mathcal{B}$. The simulator $\mathcal{B}$ plays the following game with adversary $\mathcal{A}$ on behalf of challenger $\mathcal{C}$.

Setup $\mathcal{B}$ sends the public key (g, g^a) to $\mathcal{A}$.

Phase 1 $\mathcal{A}$ accesses the encryption function E_2 many times by using arbitrary messages to ask the corresponding ciphertext. Finally, he/she outputs two messages m_1 and m_2 and sends them to $\mathcal{B}$.

Challenge $\mathcal{B}$ flips a binary coin γ and encrypts keyword m_γ as $\mathcal{E} = (C)^{H_1(m_\gamma)}$.

If $\mu = 0$, $C = g^{ab}$. As x is a private key and r is a random element in E_2, we let $a = x$ and $b = r$. Thus, we have $\mathcal{E} = (C)^{H_1(m_\gamma)} = (g^{ab})^{H_1(m_\gamma)} = g^{xrH_1(m_\gamma)}$. Therefore, $\mathcal{E}$ is a valid ciphertext of message m_γ by E_2. If $\mu = 1$, $C = g^c$. Then, we have $\mathcal{E} = g^{cH_1(m_\gamma)}$. As c is a random element, $\mathcal{E}$ is a random element in $\mathbb{G}_1$ from $\mathcal{A}$'s perspective and contains no information about m_γ.

Phase 2 $\mathcal{A}$ continues to ask the encryption oracle E.

Guess $\mathcal{A}$ outputs a guess γ' of γ. If $\gamma' = \gamma$, then $\mathcal{B}$ outputs the guess $\mu' = 0$ of μ. This means that $\mathcal{C}$ sent the valid encryption tuple $t_0 : (g, A = g^a, B = g^b, C = g^{ab})$ to $\mathcal{B}$. As $\mathcal{A}$ has the advantage ϵ to break E_2, the probability that $\mathcal{A}$ outputs guess γ' of γ satisfying $\gamma' = \gamma$ is $\frac{1}{2} + \epsilon$. Correspondingly, the probability that $\mathcal{B}$ outputs guess μ' of μ satisfying $\mu' = \mu = 0$ is $\frac{1}{2} + \epsilon$. If $\gamma' \neq \gamma$, then $\mathcal{B}$ outputs the guess $\mu' = 1$ of μ. This means that random tuple t_1 was sent to $\mathcal{B}$. Therefore, the probability that $\mathcal{A}$ outputs guess γ' of γ satisfying $\gamma' = \gamma$ is $\frac{1}{2}$. Correspondingly, the probability that $\mathcal{B}$ outputs guess μ' of μ satisfying $\mu' = \mu = 1$ is $\frac{1}{2}$.

Hence, the overall advantage that $\mathcal{B}$ solves the DDH problem can be computed as follows:

$$\left|\frac{1}{2}Pr[\mu = \mu' | \mu = 0] + \frac{1}{2}Pr[\mu = \mu' | \mu = 1] - \frac{1}{2}\right| = \left|\left[\frac{1}{2}\left(\frac{1}{2} + \epsilon\right) + \frac{1}{2} \cdot \frac{1}{2}\right] - \frac{1}{2}\right| = \frac{\epsilon}{2}$$

As ϵ is non-negligible, $\frac{\epsilon}{2}$ is also non-negligible. This conclusion means that $\mathcal{B}$ can solve the DDH problem with a non-negligible advantage, which contradicts the DDH problem assumption. □

B. SECURITY ANALYSIS

We conduct a thorough security analysis of the proposed scheme from the perspectives of requesters and workers.

- **Security of tasks**: A requester publishes his/her task, which is encrypted by AES under a symmetric key; the semantic security of AES guarantees the confidentiality of the task as long as the symmetric key is kept secret from CP.
- **Security of task encryption key**: The requester issues a task's encryption key to an authorized worker who has the ability to decrypt the task by using Lewko and Waters's multi-authority CP-AEB scheme in [1]. The collusion attack-resistant CP-AEB scheme guarantees the security of the task encryption key.

TABLE 3. Notations used in performance evaluation.

Notations	Description
P	Pairing operation
$M_{\mathbb{G}}$	Multiplication operation in group $\mathbb{G}$
$M_{\mathbb{Z}_q^*}$	Multiplication operation in $\mathbb{Z}_q^*$
$M_{v_1 \cdot v_2}$	Inner product operation between vector v_1 and vector v_2
$E_{\mathbb{G}}$	Exponentiation operation in group $\mathbb{G}$
H_1	Map a string to an element in $\mathbb{Z}_q^*$
H_2	Map a string to an element in $\mathbb{G}_1$
S_R	Attribute set administered by requester R
S_W	Attribute set of worker W (i.e., W's identities)
N_W	W's minimal attribute set satisfying an LSSS matrix, $N_W \subseteq S_W$
n	Number of rows in an LSSS matrix, each row corresponds to an attribute
$\|\alpha\|$	Dimension of α, if α is a vector; cardinality of α, if α is a set.

- **Security of task specifications**: A task specification directly reflects a task's contents, which are encrypted by encryption blocks. Our designed task specifications encryption is able to effectively resist adaptively chosen-keyword attack and chosen plaintext attack according to the proofs in **Theorem 1** and **Theorem 2**. Therefore, CP cannot gain any useful information by analyzing the encrypted task specifications under the "honest but curious" attack model.
- **Security of interests**: A worker's interests involve the worker's privacies. Our proposed interest encryption is semantically secure against chosen plaintext attack based on **Theorem 2**. Therefore, CP cannot obtain a worker's interests by inferring ciphertexts. More importantly, due to the probabilistic encryption, the same interests among workers have totally different ciphertexts, which provides stronger privacy protection for workers.

VII. EXPERIMENTAL EVALUATION

We have experimentally evaluated the proposed privacy-preserving and personalized task recommendation system. We run the proposed scheme on a Java platform based on the JPBC library [47] in a client-server environment. The client configuration is as follows: Windows 7 desktop system with 2.3-GHz Intel Core (TM) i5-6200U and 4-GB RAM; the server configuration is as follows: Ubuntu 16.04 system with 3.60-GHz Intel Core (TM) i7-7700 CPU and 8-GB RAM. The client is the requester's and the worker's work environment, which is mainly used to test task encryption and interest encryption, and the server simulate CP to perform secure task recommendation. Due to the lack of a public real-world data set on crowdsourcing tasks, we test the proposed system on a synthetic data set containing 1000 tasks, which does not affect the performance of the proposed scheme. The notations used to describe the performance of the proposed scheme are presented in Table 3. The time cost of each operation at the client side and the server side is shown in Table 4.

TABLE 4. Time cost of operation.

Operation	Client Side	Server Side
P	≈46 ms	≈11 ms
$M_{\mathbb{G}}$	<1 ms	<1 ms
$M_{\mathbb{Z}_q^*}$	<1 ms	<1 ms
$M_{v_1 \cdot v_2}$	<1 ms	<1 ms
$E_{\mathbb{G}}$	≈42 ms	≈9 ms
H_1	<1 ms	<1 ms
H_2	≈108 ms	≈26 ms

TABLE 5. Computational cost of task encryption.

Process	Computational Cost
$\mathrm{PK}_R, \mathrm{SK}_R$ Generation (**MASE.SetUp**)	$2\|S_R\|(E_{\mathbb{G}_1})$
Task Specification Encryption (**MASE.KeywordEnc**)	$P + H_1 + E_{\mathbb{G}_2} + n(2M_{v_1 \cdot v_2} + E_{\mathbb{G}_2} + 4E_{\mathbb{G}_1} + M_{\mathbb{Z}_q^*})$
Random Seed Encryption	$P + E_{\mathbb{G}_2} + M_{\mathbb{G}_2} + n(2M_{v_1 \cdot v_2} + 2E_{\mathbb{G}_2} + 3E_{\mathbb{G}_1} + M_{\mathbb{G}_1} + M_{\mathbb{G}_2})$

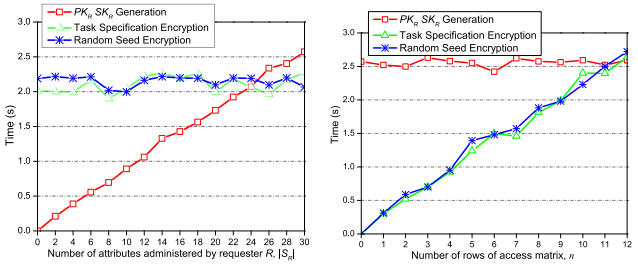
A. TASK ENCRYPTION

Recall that given the requester R's task T with task specification Q, the task encryption includes two processes: (1) use algorithm **MASE.KeywordEnc** to encrypt task specification Q and (2) use **AES** to encrypt task T. In (2), because of the high efficiency of **AES**, we only evaluate the time cost of the encryption of the random seed used to generate the **AES** symmetric key. In addition, before encrypting the task, R has to generate public key PK_R and private key SK_R by running algorithm **MASE.SetUp** based on S_R, which is a one-time operation. For the ease of reading, we have listed the theoretical computational cost of each process in Table 5. Obviously, the time complexity of $\mathrm{PK}_R, \mathrm{SK}_R$ generation is $\mathcal{O}(|S_R|)$, and the time complexity of task specification encryption and random seed encryption is $\mathcal{O}(n)$.

We have conducted the experiment in our client environment. The experimental results further demonstrated that the time cost of the public/private key generation for a requester was linear to the number of the attributes administered by the requester, while the time cost of task encryption increased linearly with the number of rows in the LSSS matrix, n (i.e., the number of attributes of task encryption). As shown in Fig. 7, the total time cost of encrypting a task was approximately $2 \times 2.189 = 4.4$s when $n = 10$.

B. WORKER ENROLLMENT

For a worker W with a set of attributes S_W, the requesters invoke algorithm **MASE.KenGen** to complete W's system enrollment on the basis of the worker's attribute set S_W. It is easy to know that the computational cost is $H_2 + 2|S_W|E_{\mathbb{G}_1}$ and the time complexity is $\mathcal{O}(|S_W|)$, which is linear to the number of attributes in S_W. When setting $|S_W| = 10$, approximately 0.95s is needed to complete the enrollment for one worker in our client environment.



(a) (b)

FIGURE 7. Time cost of task encryption. (a) For different numbers of attributes administered by a requester R with a fixed number of rows of an LSSS matrix encrypting a task T, $n = 10$. (b) For different numbers of rows of an LSSS matrix encrypting a task T with a fixed number of attributes administered by a requester R, $|S_R| = 30$.

C. INTEREST ENCRYPTION

An authorized worker W with a set of attributes S_W (the corresponding key set is SK_W) invokes algorithm **MASE.TrapdoorGen** to encrypt an interest q. Obviously, the computational cost is $H_1 + H_2 + (2 + 2|S_W|)E_{\mathbb{G}_1}$ and the time complexity is $\mathcal{O}(|S_W|)$, which is linear to the number of attributes in S_W. When setting $|S_W| = 10$, approximately 0.84s is needed to encrypt one interest in our client environment.

D. TASK RECOMMENDATION

1) PERFORMANCE EVALUATION ON OUR SCHEME

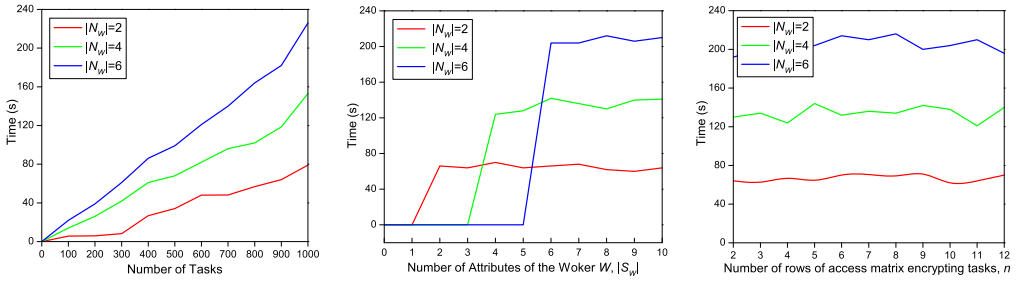
Upon receiving a worker W's trapdoor, CP runs algorithm **MASE.Match** to push suitable tasks to W. Given an encrypted task T, if, on the basis of the trapdoor, CP can find a minimal attribute set $N_W \subseteq S_W$ that satisfies the LSSS matrix embedded into the task T, then the computational cost will be $|N_W|(4M_{\mathbb{G}_2} + M_{\mathbb{G}_1} + 4P) + (|N_W| - 1)M_{\mathbb{G}_2}$, where the division x/y of two group elements x, y is equivalent to $x \times y^{-1}$ and y^{-1} is an inverse of y.

Fig. 8(a) shows that, when fixing the number of attributes of the worker W, the time cost of task recommendation linearly increases with the size of the crowdsourcing tasks. When the number of tasks is 1000, the time cost is approximately 68s, 138s, and 200s when $|N_W|$ is set to 2, 4, and 6, respectively.

Fig. 8(b) and Fig. 8(d) show that when fixing the size of tasks $t = 1000$, the number of attributes of the worker and the number of attributes used to encrypt the task (the number of rows of the LSSS matrix, n) have no influence on the time cost of the task recommendation.

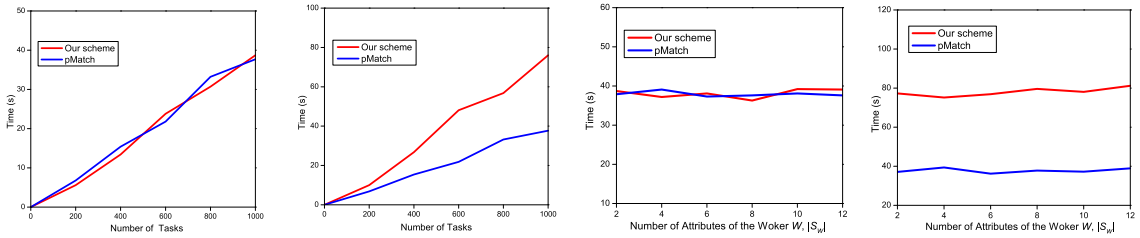
Fig. 8 shows that the time cost of task recommendation is closely related to the minimal attribute set exactly satisfying the LSSS access policy embedded into the task. The larger the number of attributes in the minimal attribute set, the higher is the time cost needed to recommend a task to the worker.

Note that in our experiments, we assume that the worker's attributes satisfy each task's access policy. When the worker's attributes do not satisfy a task's access policy, the time cost of running algorithm **MASE.Match** is almost zero, as no time-



(a) (b) (c)

FIGURE 8. Time cost of task matching. (a) For different numbers of tasks with a fixed number of attributes of the worker W, $|S_W| = 10$. (b) For different numbers of attributes of a worker with a fixed number of tasks, $t = 1000$. (c) For different numbers of attributes of encrypting tasks with a fixed number of tasks, $t = 1000$.



(a) $|N_w| = 1, |S_W| = 10$ (b) $|N_w| = 2, |S_W| = 10$ (c) $|N_w| = 1, t = 1000$ (d) $|N_w| = 2, t = 1000$

FIGURE 9. The time cost comparison between two schemes for different experimental parameters.

consuming pair operation is involved. Therefore, the actual time cost of task recommendation in a practical application is less than the experimentally obtained value.

2) PERFORMANCE COMPARISON WITH PMATCH

The time cost of task matching is a key index in the task crowdsourcing system. To provide a performance comparison with recently related work, we also implement the state-of-the-art scheme pMatch, a proxy-free privacy-preserving task matching in crowdsourcing system proposed by shu *et al.* in [9]. Fig. 9 demonstrates the time cost comparison between our scheme and pMatch. The results shows that our scheme bears approximatively the same time cost as pMatch when setting $|N_W| = 1$, while the time cost of our scheme increases linearly with the size of set N_W. When setting $|N_W| = 2$, our scheme needs twice as much time cost as pMatch. This is because that $4|N_W|$ pairing operations are required in our scheme for achieving attribute matching. Though our scheme needs more time to perform task matching, the personalized task recommendation can be achieved. How to achieve more efficient personalized task recommendation for practicability in crowdsourcing system is our next step work.

E. TASK DECRYPTION

The key operation of task decryption is that the worker W uses his/her attribute keys to recover the random seed used to generate the AES symmetric key, the computational cost of the decryption process is $(M_{\mathbb{G}_1} + 3M_{\mathbb{G}_2} + 2P)|N_W|$, where the division operation between two group elements is converted to multiplication.

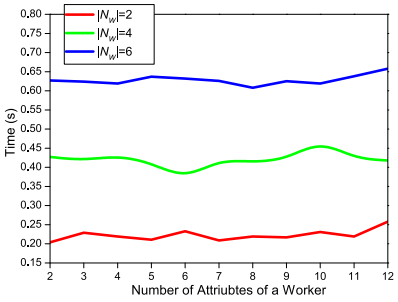


FIGURE 10. Time cost of task decryption.

Fig. 10 shows the time cost of task decryption for a worker W in our client setting for different numbers of attributes of the worker and fixed $|N_W|$ values of 2, 4, and 6. We observed that the number of attributes of the worker had no influence on the task decryption time, while the size of set N_W is linear to the task decryption time. The larger the number of attributes in N_W, the higher the time cost required to decrypt a task. When $|N_W| = 6$, the time cost of decrypting a task is approximately 0.62s.

VIII. CONCLUSION

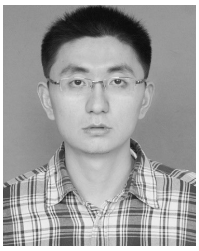
We investigate the problem of privacy-preserving and personalized task recommendation for encrypted tasks

in crowdsourcing. To achieve this goal, we first develop a new technique *multi-authority attribute-based searchable encryption* (MASE) by equipping the keyword-based searchable capability for Lewko and Waters's multi-authority CP-ABE scheme. On the basis of MASE, we further design a privacy-preserving and personalized task recommendation scheme for encrypted tasks, which allows the crowdsourcing platform to push a task to the most suitable worker. Further, we prove that the proposed scheme is secure against a ''honest but curious'' crowdsourcing platform. Finally, we show that the proposed scheme is correct and practical through extensive experiments.

REFERENCES

[1] A. Lewko and B. Waters, ''Decentralizing attribute-based encryption,'' in *Proc. EUROCRYPT*, 2011, pp. 568–588.
[2] A. Doan, R. Ramakrishnan, and A. Y. Halevy, ''Crowdsourcing systems on the World-Wide Web,'' *Commun. ACM*, vol. 54, no. 4, pp. 86–96, Apr. 2011.
[3] M. C. Yuen, I. King, and K. S. Leung, ''Task recommendation in crowdsourcing systems,'' in *Proc. ACM CrowdKDD*, 2012, pp. 22–26.
[4] D. Geiger and M. Schader, ''Personalized task recommendation in crowdsourcing information systems—Current state of the art,'' *Decision Support Syst.*, vol. 65, pp. 3–16, Sep. 2014.
[5] K. Ren, C. Wang, and Q. Wang, ''Security challenges for the public cloud,'' *IEEE Internet Comput.*, vol. 16, no. 1, pp. 69–73, Jan./Feb. 2012.
[6] S. Kamara and K. Lauter, ''Cryptographic cloud storage,'' in *Financial Cryptography and Data Security*. Berlin, Germany: Springer, Jan. 2010.
[7] J. Shu and X. Jia, ''Secure task recommendation in crowdsourcing,'' in *Proc. IEEE Global Commun. Conf.*, Dec. 2016, pp. 1–6.
[8] D. X. Song, D. Wagner, and A. Perrig, ''Practical techniques for searches on encrypted data,'' in *Proc. IEEE Symp. Secur. Privacy*, May 2000, pp. 44–55.
[9] J. Shu, K. Yang, X. Jia, X. Liu, C. Wang, and R. H. Deng, ''Proxy-free privacy-preserving task matching with efficient revocation in crowdsourcing,'' *IEEE Trans. Dependable Secure Comput.*, to be published. doi: 10.1109/TDSC.2018.2875682.
[10] J. Shu, X. Jia, K. Yang, and H. Wang, ''Privacy-preserving task recommendation services for crowdsourcing,'' *IEEE Trans. Services Comput.*, to be published. doi: 10.1109/TSC.2018.2791601.
[11] J. Shu, X. Liu, X. Jia, K. Yang, and R. H. Deng, ''Anonymous privacy-preserving task matching in crowdsourcing,'' *IEEE Internet Things J.*, vol. 5, no. 4, pp. 3068–3078, Aug. 2018.
[12] J. Shu, X. Liu, Y. Zhang, X. Jia, and R. H. Deng, ''Dual-side privacy-preserving task matching for spatial crowdsourcing,'' *J. Netw. Comput. Appl.*, vol. 123, pp. 101–111, Dec. 2018.
[13] J. Shu, X. Liu, K. Yang, Y. Zhang, X. Jia, and R. H. Deng, ''SybSub: Privacy-preserving expressive task subscription with sybil detection in crowdsourcing,'' *IEEE Internet Things J.*, vol. 6, no. 2, pp. 3003–3013, Apr. 2019.
[14] V. Ambati, S. Vogel, and J. G. Carbonell, ''Towards task recommendation in micro-task markets,'' in *Proc. AAAI*, 2011, pp. 1–4.
[15] D. E. Difallah, G. Demartini, and P. Cudré-Mauroux, ''Pick-a-crowd: Tell me what you like, and i'll tell you what to do,'' in *Proc. ACM WWW*, 2013, pp. 367–374
[16] A. Sahai and B. Waters, ''Fuzzy identity-based encryption,'' in *Proc. EUROCRYPT*, 2005, pp. 457–473.
[17] E.-J. Goh. (2003). Secure Indexes. IACR ePrint Cryptography Archive. [Online]. Available: http://eprint.iacr.org/2003/216
[18] Y.-C. Chang and M. Mitzenmacher, ''Privacy preserving keyword searches on remote encrypted data,'' in *Proc. ACNS*. Berlin, Germany: Springer, 2005, pp. 442–455.
[19] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, ''Searchable symmetric encryption: Improved definitions and efficient constructions,'' in *Proc. ACM CCS*, vol. 19, 2006, pp. 79–88.
[20] S. Kamara, C. Papamanthou, and T. Roeder, ''Dynamic searchable symmetric encryption,'' in *Proc. ACM CCS*, 2012, pp. 965–976.
[21] E. Stefanov, C. Papamanthou, and E. Shi, ''Practical dynamic searchable encryption with small leakage,'' in *Proc. NDSS*, 2014, pp. 72–75.
[22] R. Bost, ''$\Sigma o \varphi o \varsigma$: Forward secure searchable encryption,'' in *Proc. ACM CCS*, 2016, pp. 1143–1154.
[23] K. S. Kim, M. Kim, D. Lee, J. H. Park, and W.-H. Kim, ''Forward secure dynamic searchable symmetric encryption with efficient updates,'' in *Proc. ACM CCS*, 2017, pp. 1449–1463.
[24] K. Kurosawa and Y. Ohtaki, ''Uc-secure searchable symmetric encryption,'' in *Financial Cryptography and Data Security*. Berlin, Germany: Springer, 2012, pp. 285–298.
[25] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, ''Public key encryption with keyword search,'' in *Proc. EUROCRYPT*, 2004, pp. 506–522.
[26] P. Golle, J. Staddon, and B. Waters, ''Secure conjunctive keyword search over encrypted data,'' in *Proc. ACNS*. Berlin, Germany: Springer, 2004, pp. 31–45.
[27] L. Ballard, S. Kamara, and F. Monrose, ''Achieving efficient conjunctive keyword searches over encrypted data,'' in *Proc. IEEE ICICS*, Dec. 2005, pp. 414–426.
[28] D. Boneh and B. Waters, ''Conjunctive, subset, and range queries on encrypted data,'' in *Proc. TCC*. Berlin, Germany: Springer, 2007, pp. 535–554.
[29] V. Goyal, O. Pandey, A. Sahai, and B. Waters, ''Attribute-based encryption for fine-grained access control of encrypted data,'' in *Proc. ACM CCS*, 2006, pp. 89–98.
[30] R. Ostrovsky, A. Sahai, and B. Waters, ''Attribute-based encryption with non-monotonic access structures,'' in *Proc. ACM CCS*, 2007, pp. 195–203.
[31] J. Bethencourt, A. Sahai, and B. Waters, ''Ciphertext-policy attribute-based encryption,'' in *Proc. IEEE Symp. Secur. Privacy*, May 2007, pp. 321–334.
[32] L. Cheung and C. Newport, ''Provably secure ciphertext policy ABE,'' in *Proc. ACM CCS*, 2007, pp. 456–465.
[33] L. Ibraimi, Q. Tang, P. Hartel, and W. Jonker, ''Efficient and provable secure ciphertext-policy attribute-based encryption schemes,'' in *Proc. Int. Conf. Inf. Secur. Pract. Exper.*, 2009, pp. 1–12.
[34] S. Wang, J. Ye, and Y. Zhang, ''A keyword searchable attribute-based encryption scheme with attribute update for cloud storage,'' *PLoS ONE*, vol. 13, no. 5, 2018, Art. no. e0197318.
[35] S. Wang, D. Zhang, Y. Zhang, and L. Liu, ''Efficiently revocable and searchable attribute-based encryption scheme for mobile cloud storage,'' *IEEE Access*, vol. 6, pp. 30444–30457, 2018.
[36] S. Wang, L. Yao, and Y. Zhang, ''Attribute-based encryption scheme with multi-keyword search and supporting attribute revocation in cloud storage,'' *PLoS ONE*, vol. 13, no. 10, 2018, Art. no. e0205675.
[37] S. Wang, K. Guo, and Y. Zhang, ''Traceable ciphertext-policy attribute-based encryption scheme with attribute level user revocation for cloud storage,'' *PLoS ONE*, vol. 13, no. 9, 2018, Art. no. e0203225.
[38] T. Peng, L. Qin, B. Hu, J. Liu, and J. Zhu, ''Dynamic keyword search with hierarchical attributes in cloud computing,'' *IEEE Access*, vol. 6, pp. 68948–68960, 2018.
[39] H. Yin, J. Zhang, Y. Xiong, L. Ou, F. Li, S. Liao, and K. Li, ''CP-ABSE: A ciphertext-policy attribute-based searchable encryption scheme,'' *IEEE Access*, vol. 7, pp. 5682–5694, 2019.
[40] L. B. Chilton, J. J. Horton, R. C. Miller, and S. Azenkot, ''Task search in a human computation market,'' in *Proc. ACM SIGKDD*, 2010, pp. 1–9.
[41] M.-C. Yuen, I. King, and K.-S. Leung, ''TaskRec: Probabilistic matrix factorization in task recommendation in crowdsourcing systems,'' in *Proc. ICONIP*. Berlin, Germany: Springer, 2012, pp. 516–525.
[42] B. Ye and Y. Wang, ''CrowdRec: Trust-aware worker recommendation in crowdsourcing environments,'' in *Proc. IEEE ICWS*, Jun./Jul. 2016, pp. 1–8.
[43] K. Yang, K. Zhang, J. Ren, and X. Shen, ''Security and privacy in mobile crowdsourcing networks: Challenges and opportunities,'' *IEEE Commun. Mag.*, vol. 53, no. 8, pp. 75–81, Aug. 2015.
[44] H. Xiong and Z. Qin, ''Revocable and scalable certificateless remote authentication protocol with anonymity for wireless body area networks,'' *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 7, pp. 1442–1455, Jul. 2015.
[45] G. Meiselwitz and J. Lazar, ''Accessibility of registration mechanisms in social networking sites,'' in *Proc. OCSC*. Berlin, Germany: Springer, 2009, pp. 82–90.
[46] J. Katz and Y. Lindell, *Introduction to Modern Cryptography*. Boca Raton, FL, USA: CRC Press, 2007.
[47] Accessed: Jun. 18, 2019. [Online]. Available: http://gas.dia.unisa.it/projects/jpbc/index.html

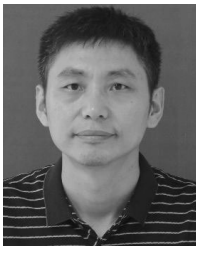
HUI YIN received the B.S. degree in computer science from Hunan Normal University, China, in 2002, the M.S. degree in computer software and theory from Central South University, China, in 2008, and the Ph.D. degree from the College of Information Science and Engineering, Hunan University, China, in 2018. He is currently an Assistant Professor with the College of Applied Mathematics and Computer Engineering, Changsha University, China. His research interests include information security, privacy protection, and applied cryptography.



YINQIAO XIONG received the B.S. and M.S. degrees in computer science and technology from the School of Computer, National University of Defense Technology (NUDT), Changsha, China, in 2007 and 2010, respectively, where he is currently pursuing the Ph.D. degree in cyberspace security with the School of Computer. He is also a Lecturer with Changsha University. His research interests include privacy preserving, information security, and the Internet of Things.

TIANTIAN DENG received the M.S. degree in software engineering from the School of Software, Sun Yat-sen University, Guangzhou, China, in 2005. She is currently pursuing the Ph.D. degree in software engineering with the School of Computer, National University of Defense Technology, Changsha, China. She is also a Senior Engineer with Changsha University. Her research interests include big data analysis and open source ecology.

HUA DENG received the M.S. degree in cryptography from Southwest Jiaotong University, China, in 2010, and the Ph.D. degree in information security from Wuhan University, China, in 2015. He is currently a Postdoctoral Research Fellow with the College of Computer Science and Electronic Engineering, Hunan University, China. His research interests include applied cryptography, data security and privacy, and cloud security.



PEIDONG ZHU received the Ph.D. degree in computer science from the National University of Defense Technology (NUDT), in 1999, where he was a Professor with the College of Computer, until 2017. He is currently with the College of Electronic Information and Electrical Engineering, Changsha University, China. His research interests include security of large-scale cyber-physical networks and architecture of the Internet.