Hindawi
Security and Communication Networks
Volume 2020, Article ID 1238505, 11 pages
https://doi.org/10.1155/2020/1238505

Research Article

Highly Secure Privacy-Preserving Outsourced *k*-Means Clustering under Multiple Keys in Cloud Computing

Ying Zou,[1,2] Zhen Zhao,[3] Sha Shi,[4] Lei Wang,[1] Yunfeng Peng,[5] Yuan Ping,[6] and Baocang Wang[3]

[1]*Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China*
[2]*Department of Mathematics Teaching and Research, Shanghai Business School, Shanghai, China*
[3]*State Key Laboratory of Integrated Service Networks, Xidian University, Xi'an, China*
[4]*Engineering Research Center of Molecular and Neuro Imaging of Ministry of Education of China and School of Life Science and Technology, Xidian University, Xi'an, China*
[5]*PBC School of Finance, Tsinghua University, Beijing, China*
[6]*School of Information Engineering, Xuchang University, Xuchang, China*

Correspondence should be addressed to Baocang Wang; bcwang79@aliyun.com

Received 13 November 2019; Accepted 5 December 2019; Published 9 March 2020

Academic Editor: Prosanta Gope

Data clustering is the unsupervised classification of data records into groups. As one of the steps in data analysis, it has been widely researched and applied in practical life, such as pattern recognition, image processing, information retrieval, geography, and marketing. In addition, the rapid increase of data volume in recent years poses a huge challenge for resource-constrained data owners to perform computation on their data. This leads to a trend that users authorize the cloud to perform computation on stored data, such as keyword search, equality test, and outsourced data clustering. In outsourced data clustering, the cloud classifies users' data into groups according to their similarities. Considering the sensitive information in outsourced data and multiple data owners in practical application, it is necessary to develop a privacy-preserving outsourced clustering scheme under multiple keys. Recently, Rong et al. proposed a privacy-preserving outsourced *k*-means clustering scheme under multiple keys. However, in their scheme, the assistant server (AS) is able to extract the ratio of two underlying data records, and key management server (KMS) can decrypt the ciphertexts of owners' data records, which break the privacy security. AS can even reduce all data records if it knows one of the data records. To solve the aforementioned problem, we propose a highly secure privacy-preserving outsourced *k*-means clustering scheme under multiple keys in cloud computing. In this paper, noncolluded cloud computing service (CCS) and KMS jointly perform clustering over the encrypted data records without exposing data privacy. Specifically, we use BCP encryption which has additive homomorphic property and AES encryption to double encrypt data records, where the former cryptosystem prevents CCS from obtaining any useful information from received ciphertexts and the latter one protects data records from being decrypted by KMS. We first define five protocols to realize different functions and then present our scheme based on these protocols. Finally, we give the security and performance analyses which show that our scheme is comparable with the existing schemes on functionality and security.

1. Introduction

Data clustering [1, 2] enables data records to be classified into groups according to their features, attributes, or similarities. This property leads to its significance in many fields related to data analysis, such as pattern recognition, image processing, information retrieval, geography, and marketing. Furthermore, with the explosive data received nowadays in the information era, it has been a challenge for our digital devices not only to storage but also to perform computation on such massive data. Cloud computing relieves this problem by providing a platform with high storage capacity and strong computing power. Users tend to outsource their data on the cloud and authorize the cloud server computing ability on data. The cloud server therefore can replace users to perform some computation on the outsourced data, such as keyword search [3], equality test [4], and outsourced data clustering [5]. It is worth noting that, in these applications, the cloud server will send the final result to the data owner. This gives a security issue of data integrity which has been further researched in [6–11].

By outsourced data clustering which means the cloud classifies data into different groups according to their similarities, it is possible to efficiently detect abnormalities, segment images, and predict diseases. As a widely applied clustering method, k-means clustering [1] classifies data into k-clusters based on their distances from cluster centers. However, the sensitive information of data on the cloud platform cannot be protected by simply using k-means clustering. This calls for privacy-preserving outsourced k-means clustering, where data is classified without exposing the sensitive information of data.

The traditional privacy-preserving k-means clustering schemes [12–15] protect the data privacy by adding noises with the sacrifice of clustering accuracy. Subsequently, some symmetric and asymmetric constructions [16–18] have been proposed to improve it with the tradeoff of computing cost and communication overhead. The literature of outsourced privacy-preserving clustering schemes fall into two categories, i.e., single-key and multikey clustering, where the former one refers to that all of the outsourced data of owners are encrypted with one same key while that are with different keys in multikey clustering. Taking into account the practical application, it is necessary to consider the privacy-preserving clustering under multiple keys.

Recently, Rong et al. proposed an outsourced k-means clustering scheme [19] under multiple keys. Nevertheless, their scheme is not secure against semihonest assistant server (AS) and key management server (KMS), where AS can extract the ratio of messages and KMS can even extract all data records of users with its master secret key. In addition, as long as AS obtains one of the data records, it can recover all data records. The privacy leakage may incur a huge economic loss to the user in practice. To solve this problem, we present a highly secure outsourced k-means clustering scheme under multiple keys in cloud computing.

1.1. Our Contribution. In this paper, we propose a highly secure privacy-preserving outsourced k-means clustering under multiple keys in cloud computing.

We first introduce our system model and threat models. Specifically, the system model includes four entities, i.e., data owners (DOs), query client (QC), cloud computing service (CCS), and key management service (KMS), and threat model denotes the models against semihonest CCS and KMS. Subsequently, based on [19] and BCP homomorphic encryption, we construct five protocols to realize different functions. It is worth noting that the secure multiplication (SM) protocol is defined to achieve the multiplicative homomorphic property using BCP encryption which only has additive homomorphic property. We then present a highly secure outsourced k-means clustering scheme under multiple keys in cloud computing, which achieves privacy security against semihonest CCS and KMS. In particular, we use BCP encryption to realize the security against privacy leakage to CCS such that semihonest CCS cannot extract any useful information from ciphertexts of data records. We then utilize AES encryption to protect privacy security against semihonest KMS. KMS, therefore, cannot extract any data records of data owners although KMS possesses the master secret key which can be used to decrypt ciphertexts encrypted using BCP encryption.

1.2. Related Work

1.2.1. Privacy-Preserving k-Means Clustering. Zhang et al. [20] proposed a high-order possibilistic c-means algorithm for big data in cloud computing based on the BGV cryptosystem [21]. However, their scheme is not practical because of its low efficiency. Subsequently, Almutairi et al. [22] improved it and developed a privacy-preserving k-means clustering scheme based on homomorphic encryption but failed to protect the plaintext information in the update of clustering centers. For this, Yuan and Tian [23] put forward a privacy-preserving clustering scheme using a novel lightweight cryptosystem basing on the hardness of learning with error (LWE) [24]. Their scheme can complete the sum of ciphertexts and compare the distance using ciphertexts of multidimensional data. Nevertheless, this scheme is not fully outsourced.

1.2.2. Outsourced Single-Key Clustering. Lin [25] constructed a privacy-preserving kernel k-means clustering scheme based on linear transformation and kernel matrix with random perturbation, but this scheme cannot realize ciphertext comparison. Based on Paillier cryptosystem, Rao et al. [26] proposed a privacy-preserving outsourcing distributed clustering protocol in the union cloud environment, which includes a new protocol to construct the function of Euclidean distance and evaluate the termination condition over the encrypted data. The problem of this scheme lies in the heavy computing load and lack of support to encrypted datasets under multiple keys. Liu et al. [27] constructed a secure KNN multilabel data classification scheme based on Paillier cryptosystem.

1.2.3. Multikey Clustering. Gheid and Challal [28] presented a novel privacy-preserving k-means clustering scheme with the multiparty of Clifton security [29]. Peter et al. [30] further proposed a scheme to outsource multiparty computation to cloud under multiple keys, while it does not support ciphertext comparison. Li et al. [31] applied the BCP homomorphic encryption [32] to multiparty horizontal partitioned databases and then set up the ciphertext comparison for the outsourced privacy-preserving random decision tree algorithm. Rong et al. [19] improved it by presenting an efficient privacy-preserving protocol for outsourced k-means clustering under multiple keys based on the double decryption cryptosystem [33].

1.3. Organization. The rest of this paper is organized as follows. In Section 2, we recall the definitions for k-means clustering, BCP encryption, and AES encryption. The system model and threat models are proposed in Section 3. In Section 4, five basic protocols are constructed, and we present our scheme in which the defined protocols are invoked thoroughly. The security proof and performance analysis are given in Section 5. Finally, we conclude this paper in Section 6.

2. Preliminaries

2.1. Notations. We summarize the notations used in this paper in Table 1.

2.2. k-Means Clustering. k-means clustering is an iterative algorithm that allocates l data records into k disjoint clusters, each of which has a center. Let l m-dimensional data records be $\vec{d}_1, \vec{d}_2, \ldots, \vec{d}_l$ and k clusters be $c_1, c_2, \ldots, c_k$, where $\vec{\mu}_1, \vec{\mu}_2, \ldots, \vec{\mu}_k$ are the centers of k clusters separately. The data record $\vec{d}_i$ will be categorized into the cluster c_j if $\vec{d}_i$ and $\vec{\mu}_j$ has the minimum Euclidean distance among that of $\vec{d}_i$ and all of cluster centers. In particular, the Euclidean distance of an m-dimensional data record $\vec{d}_i = (d_{i,1}, d_{i,2}, \ldots, d_{i,m})$ and a cluster center $\vec{\mu}_j = (\mu_{j,1}, \mu_{j,2}, \ldots, \mu_{j,m})$ can be expressed as

$$\mathrm{Dist}\left(\vec{d}_i, \vec{\mu}_j\right) = \sum_{h=1}^{m} \left(d_{i,h} - \mu_{j,h}\right)^2. \quad (1)$$

The detailed process of k-means clustering is depicted as Algorithm 1. The algorithm takes as input l m-dimensional data records $\vec{d}_1, \vec{d}_2, \ldots, \vec{d}_l$, a predefined number of clusters k, and a predefined max number of iterations I. k-cluster centers are firstly picked to compute the Euclidean distance with data records. Each data record is distributed to the cluster which has the minimum Euclidean distance with it. After one iteration, the cluster center $\vec{\mu}_j$ is reassigned as the average value of all data records in c_j for $j \in \{1, 2, \ldots, k\}$. If the max number of iterations is reached or the output clusters does not change any more, terminate the algorithm and output the k-clusters.

2.3. BCP Encryption. In this paper, we utilize the BCP encryption proposed by Bresson et al. [32] which has the additive homomorphic property and provides double decryption mechanisms. The BCP encryption consists of five algorithms as follows:

(i) Setup(λ). Taking as input a security parameter λ, the setup algorithm picks two primes p, q of the form $p = 2p' + 1, q = 2q' + 1$ and computes $N = pq$, where p', q' are also primes. Consider $\mathbb{G} = QR_{N^2}$, the cyclic group of quadratic residues modulo N^2, and we have ord $(\mathbb{G}) = N\lambda(N)/2$ with $\lambda(N) = 2p'q'$. It chooses $g \in \mathbb{G}$, the order of which is $N\lambda(N)/2$, and we have $g^{\lambda(N)} \bmod N^2 = (1 + \alpha N) \bmod N^2$, $\alpha \in \{1, 2, \ldots, N-1\}$. The public parameter pp and the master secret key msk are denoted as

Table 1: Notations.

Symbol	Meaning
K	Number of clusters
L	Number of total data records
m	Dimension of data records
I	Maximum number of iterations
n	Number of data owners
n_i	Number of data records of i-th owner
$\vec{d}_i$	i-th data record $\vec{d}_i = (d_{i,1}, d_{i,2}, \ldots, d_{i,m})$ with $i \in [1, l]$
c_i	i-th cluster with $i \in [1, k]$
$\lvert c_i \rvert$	Number of data records in cluster c_i
$\vec{\mu}_i$	Cluster center of i-th cluster
$\mathrm{Dist}(\vec{a}, t\,\vec{b})$	Euclidean distance between vectors $\vec{a}$ and $\vec{b}$
DO_i	i-th data owner with $i \in [1, n]$
QC	Query client
CCS	Cloud computing service
KMS	Key management server
λ	The security parameter
p, q	Two primes of the form $p = 2p' + 1, q = 2q' + 1$
$N, \lambda(N)$	$N = pq, \lambda(N) = 2p'q'$
$\mathbb{G}$	The cyclic group of quadratic residue modolo N^2
g	$g \in \mathbb{G}$ and its order is $N\lambda(N)/2$
msk	Master secret key in BCP encryption
pk, sk	Public key and secret key in BCP encryption
ask	Symmetric key used in AES encryption
Λ	$\Lambda = (M_1 + r_1)(M_2 + r_2)$
$\Omega_{i,j}$	Scaled squared distance between $\vec{d}_i$ and $\vec{\mu}_j$
$V_{n\times k}$	Location matrix of n records in k clusters

$$\mathrm{pp} = (N, g), \mathrm{msk} = (p', q'). \quad (2)$$

(ii) KeyGen(pp). Taking as input the public parameter pp, the key generation algorithm randomly chooses $a \in [1, \mathrm{ord}(\mathbb{G})]$ and computes $h = g^a \bmod N^2$. Note that h is of maximal order with high probability. It sets the output public and secret key pair (pk, sk) as

$$\mathrm{pk} = h = g^a, \mathrm{sk} = a. \quad (3)$$

(iii) Enc(pk, M). Taking as input a public key pk and a message M, the encryption algorithm randomly chooses $r \in \mathbb{Z}_N$ and generates the ciphertext $\mathrm{CT} = (A, B)$ as

$$A = g^r \bmod N^2, B = h^r (1 + mN) \bmod N^2. \quad (4)$$

Specifically, we denote $\mathrm{Enc}_{\mathrm{pk}}(M)$ as the encryption of message M under the public key pk.

(iv) Dec$(\mathrm{sk}, \mathrm{CT})$. Taking as input a secrete key sk and a ciphertext $\mathrm{CT} = (A, B)$, the decryption algorithm output the message as

$$M = \frac{B/A^a - 1 \bmod N^2}{N}. \quad (5)$$

Specifically, we denote $\mathrm{Dec}_{\mathrm{sk}}(\mathrm{CT})$ as the decryption of ciphertext CT under the secret key sk.

Input: $\overrightarrow{d}_1, \overrightarrow{d}_2, \ldots, \overrightarrow{d}_l$: m-dimensional data records, where $\overrightarrow{d}_i = (d_{i,1}, d_{i,2}, \ldots, d_{i,m})$;
 k: predefined number of clusters;
 I: predefined max number of iterations;
Begin: Pick k cluster centers $\overrightarrow{\mu}_j$, $j \in \{1, 2, \ldots, k\}$,
 for $\alpha = 1$ to I **do**
 if {different k clusters compared with the case $\alpha - 1$}
 (1) Distribute each data record $\overrightarrow{d}_i$ to the cluster c_j with the minimum $\mathrm{Dist}(\overrightarrow{d}_i, \overrightarrow{\mu}_j)$ among $\mathrm{Dist}(\overrightarrow{d}_i, \overrightarrow{\mu}_h)$ for $h \in \{1, 2, \ldots, k\}$
 (2) Update the cluster center $\overrightarrow{\mu}_j$ to the average values of data records in c_j for $i \in \{1, 2, \ldots, k\}$, $\alpha = \alpha + 1$
 end if
 end for
Output: k clusters

Algorithm 1: k-means clustering.

(v) $\mathrm{sDec}(\mathrm{msk}, \mathrm{CT})$. Taking as input the master secret key msk and a ciphertext $\mathrm{CT} = (A, B)$, the system decryption algorithm computes

$$a \bmod N = \frac{h^{\lambda(N)} - 1 \bmod N^2}{N} \cdot \alpha^{-1} \bmod N,$$
$$r \bmod N = \frac{A^{\lambda(N)} - 1 \bmod N^2}{N} \cdot \alpha^{-1} \bmod N. \quad (6)$$

Let $ar\mathrm{ord}(\mathbb{G}) = \gamma_1 + \gamma_2 N$; thus, $ar = \gamma_1 \bmod N$ is efficiently computable. Let π be the inverse of λ_N. It generates the message as

$$M = \frac{\left(B/g^{\gamma_1}\right)^{\lambda(N)} - 1 \bmod N^2}{N} \cdot \pi \bmod N. \quad (7)$$

Specifically, we denote $\mathrm{sDec}_{\mathrm{msk}}(\mathrm{CT})$ as the decryption of ciphertext CT under the master secret key msk.

Specifically, BCP encryption has additive homomorphic property, which means

$$\mathrm{Enc}(M_1) \cdot \mathrm{Enc}(M_2) = \mathrm{Enc}(M_1 + M_2). \quad (8)$$

This property will be utilized in the whole system.

2.4. AES Encryption. AES encryption is an efficient symmetric encryption system widely used in practical application, where the symmetric means encryption and decryption require the same key. We give the simplified definition of AES as follows:

(i) AKeyGen. The sender and receiver consult the secret key sk of the AES encryption system.

(ii) AEnc. The sender generates the ciphertext CT of message M under the secret key ask following the AES encryption algorithm. We denote it as

$$\mathrm{CT} = \mathrm{AEnc}_{\mathrm{ask}}(M). \quad (9)$$

(iii) ADec. The receiver decrypts the ciphertext CT with the secret key sk. We denote it as

$$M = \mathrm{Dec}_{\mathrm{ask}}(\mathrm{CT}). \quad (10)$$

3. Models

3.1. System Model. As shown in Figure 1, our scheme considers four types of entities, i.e., data owner (DO), cloud computing service (CCS), key management server (KMS), and query client (QC).

(i) DO: DO has limited computing power and therefore outsources its encrypted data to the cloud. Our system involves n DOs, denoted as $\mathrm{DO}_1, \mathrm{DO}_2, \ldots, \mathrm{DO}_n$. For $i \in [1, n]$, each DO_i has n_i data records, and each data record has m attributes. Data owners are assumed not to collude with the cloud servers.

(ii) QC: QC is authorized to query and receive the clustering results and does not involve in any clustering calculation.

(iii) CCS: CCS stores the datasets of multiple DOs, takes part in the clustering process, and sends the clustering results to the QC.

(iv) KMS: KMS generates system parameters and performs ciphertext transformation with the master secret key. It also participates in the clustering process.

3.2. Threat Models. In our system, we suppose that CCS and KMS are semihonest. This means they will honestly perform what the protocol requires but will be curious about the messages under ciphertexts they received. Upon this assumption, we define three thread models as follows, where an adversary $\mathscr{A}$ acting as different roles in different models attempts to decrypt the ciphertexts sent from DOs and CCS.

(i) Acting as a "malicious" CCS, $\mathscr{A}$ tries to obtain the message under ciphertexts sent from DOs and KMS

(ii) Acting as a "malicious" KMS, $\mathscr{A}$ tries to obtain the real message under ciphertexts sent from CCS

(iii) Acting as a "malicious" KMS, $\mathscr{A}$ tries to obtain the message under the ciphertexts that sent from DOs to CCS

It is worth noting that CCS and KMS are assumed not to collude with each other.

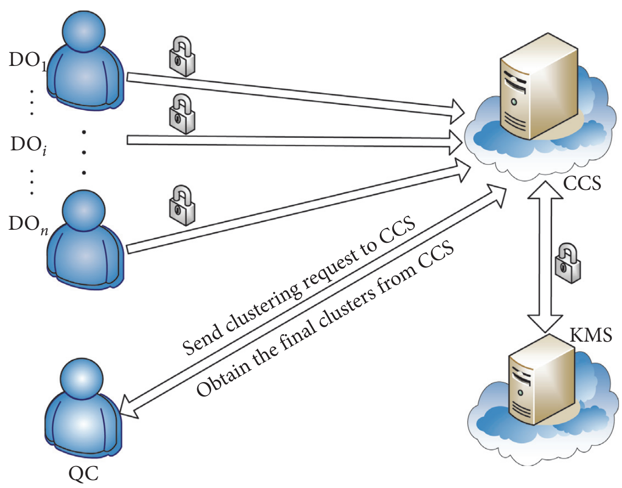

Figure 1: System architecture.

4. Our Construction

Based on the scheme proposed by Rong et al. in [19], we construct a more secure clustering scheme. In our construction, we utilize BCP homomorphic encryption to protect the privacy security of data owners such that adversaries cannot extract any useful information about underlying data records of data owners, while AS can easily extract M_1/M_2 in [19]. Furthermore, AES encryption is also used to double-encrypt the data records to prevent KMS from directly extracting data records from ciphertexts sent from DO to CCS.

4.1. Protocols. We first define five underlying protocols to satisfy different requirements in the clustering process. To securely transfer the data records of DO to CCS, we define secure ciphertext transformation (SCT) protocol. Since the BCP encryption used in our scheme only has additive property, we build a secure multiplication (SM) protocol to realize the multiplicative property. Finally, aiming to classify the similar data records using the ciphertexts, we construct three protocols, namely, secure distance measurement (SDM) protocol, secure distance comparison (SDC) protocol, and secure minimum distance measurement (SMDM) protocol. These protocols will be invoked through our scheme.

4.1.1. Secure Ciphertext Transformation Protocol. Secure ciphertext transformation (SCT) protocol aims to transfer the ciphertext of message M encrypted under public key pk_x to a ciphertext of M encrypted under public key pk_y without revealing M. Suppose two entities in SCT protocol, i.e., Alice and Bob, Alice interacts with Bob following SCT protocol to convert $\mathrm{Enc}_{\mathrm{pk}_x}(M)$ to $\mathrm{Enc}_{\mathrm{pk}_y}(M)$. To prevent Bob from extracting the message M, a random number is used to blind the message from Bob. The detailed process is listed in Algorithm 2.

Taking as the input the public keys pk_x and pk_y and the ciphertext $\mathrm{Enc}_{\mathrm{pk}_x}(M)$, Alice randomly chooses $r \in \mathbb{Z}_N$ and encrypts r using pk_x to $\mathrm{Enc}_{\mathrm{pk}_x}(r)$. It then computes the encryption of $(M+r)$ under pk_x, which can be realized by $\mathrm{Enc}_{\mathrm{pk}_x}(M) + \mathrm{Enc}_{\mathrm{pk}_x}(r)$ because of the additive homomorphic property of BCP encryption. Alice then sends the output sf $\mathrm{Enc}_{\mathrm{pk}_x}(M+r)$ to Bob. Taking as the input the public key pk_y, its master secret key msk, and received $\mathrm{Enc}_{\mathrm{pk}_x}(M+r)$, Bob decrypts this ciphertext using its master secret key msk following the system decryption algorithm sDec and obtains $(M+r)$. It then encrypts $(M+r)$ with pk_y and sends the output $\mathrm{Enc}_{\mathrm{pk}_y}(M+r)$ to Alice. Alice eliminates r in the ciphertext by computing $\mathrm{Enc}_{\mathrm{pk}_y}(M+r)\cdot \mathrm{Enc}_{\mathrm{pk}_y}(-r)$ and obtains $\mathrm{Enc}_{\mathrm{pk}_y}(M)$ as the final output.

4.1.2. Secure Multiplication Protocol. Secure multiplication (SM) protocol is used to obtain the ciphertext of messages' multiplication with corresponding messages' ciphertexts using the BCP homomorphic cryptosystem. It is required in this process that the messages should not be exposed. The same as SCT protocol, we also assume two entities in SM protocol, i.e., Alice and Bob. Alice attempts to obtain $\mathrm{Enc}_{\mathrm{pk}_x}(M_1\cdot M_2)$ from $\mathrm{Enc}_{\mathrm{pk}_x}(M_1), \mathrm{Enc}_{\mathrm{pk}_x}(M_2)$ without revealing M_1, M_2 to Bob who is the owner of the corresponding secret key sk. We define SM protocol in Algorithm 3.

Taking as the input the ciphertext $\mathrm{Enc}_{\mathrm{pk}_x}(M_1)$ and $\mathrm{Enc}_{\mathrm{pk}_y}(M_2)$, Alice randomly chooses numbers $r_1, r_2 \in \mathbb{Z}_N$ and computes the ciphertext of $(M_1+r_1), (M_2+r_2)$ by computing $\mathrm{Enc}_{\mathrm{pk}_x}(M_1)\cdot \mathrm{Enc}_{\mathrm{pk}_x}(r_1)$ and $\mathrm{Enc}_{\mathrm{pk}_x}(M_2)\cdot \mathrm{Enc}_{\mathrm{pk}_x}(r_2)$ respectively. This utilizes the additive homomorphic property of BCP encryption. It then sends the output $\mathrm{Enc}_{\mathrm{pk}_x}(M_1+r_1)$, $\mathrm{Enc}_{\mathrm{pk}_x}(M_2+r_2)$ to Bob. Taking as the input the corresponding secret key sk_x of pk_x, Bob decrypts the received ciphertexts with sk_x and obtains (M_1+r_1), (M_2+r_2). It computes the multiplication of (M_1+r_1), (M_2+r_2) as $\Lambda = (M_1+r_1)\cdot(M_2+r_2) = M_1\cdot M_2 + r_2M_1 + r_1M_2 + r_1r_2$ and encrypts Λ with pk_x as $\mathrm{Enc}_{\mathrm{pk}_x}(\Lambda)$ which is used to divide $M_1\cdot M_2$ in the underlying message. Bob sends $\mathrm{Enc}_{\mathrm{pk}_x}(\Lambda)$ to Alice. Finally, Alice computes $\mathrm{Enc}_{\mathrm{pk}_x}(M_1\cdot M_2)$ with $\mathrm{Enc}_{\mathrm{pk}_x}(\Lambda), r_1, r_2, \mathrm{Enc}_{\mathrm{pk}_x}(M_1), \mathrm{Enc}_{\mathrm{pk}_x}(M_2)$ using the additive homomorphic property of BCP encryption.

4.1.3. Secure Distance Measurement Protocol. We define the secure distance measurement (SDM) protocol to measure the distance between data records and cluster centers using Euclidean distance. Assume there are n data records and k clusters. Let $\vec{s}_j = (s_{j,1}, s_{j,2}, \ldots, s_{j,m})$ be the sum of data records in cluster c_j and $|c_j|$ be the number of data records in cluster c_j, respectively. Given a data record $\vec{d}_i = (d_{i,1}, d_{i,2}, \ldots, d_{i,m})$ and a cluster center $\vec{\mu}_j = (\mu_{i,1}, \mu_{i,2}, \ldots, \mu_{i,m})$, $\Omega_{i,j}$ is denoted as the scaled squared distance between $\vec{d}_i$ and $\vec{\mu}_j$ satisfying $\sqrt{\mathrm{Dist}(\vec{d}_i, \vec{\mu}_j)} = \sqrt{\Omega_{i,j}}/|c_j|$. Therefore, $\Omega_{i,j}$ is denoted and computed as in the following equation:

$$\begin{aligned}\Omega_{i,j} &= \left(\sqrt{\mathrm{Dist}\left(\vec{d}_i, \vec{\mu}_j\right)}\cdot\left|c_j\right|\right)^2. \\ &= \sum_{\alpha=1}^{m}\left(\left|c_j\right|, d_{i,\alpha} - s_{j,\alpha}\right)^2\end{aligned} \tag{11}$$

The process is depicted as Algorithm 4.

Input: Alice: $\mathrm{pk}_x, \mathrm{pk}_y, \mathrm{Enc}_{\mathrm{pk}_x}(M)$
Bob: msk, pk_y
Begin: Alice:
(a) Pick a random number $r \in \mathbb{Z}_N$
(b) Compute $\mathrm{Enc}_{\mathrm{pk}_x}(M+r) \longleftarrow \mathrm{Enc}_{\mathrm{pk}_x}(M) \cdot \mathrm{Enc}_{\mathrm{pk}_y}(r)$
(c) Send $\mathrm{Enc}_{\mathrm{pk}_x}(M+r)$ to Bob
Bob:
(a) Decrypt $(M+r) \longleftarrow \mathrm{sDec}_{\mathrm{msk}}(\mathrm{Enc}_{\mathrm{pk}_x}(M+r))$
(b) Compute $\mathrm{Enc}_{\mathrm{pk}_y}(M+r)$
(c) Send $\mathrm{Enc}_{\mathrm{pk}_y}(M+r)$ to Alice
Alice
(a) Compute $\mathrm{Enc}_{\mathrm{pk}_y}(M) = \mathrm{Enc}_{\mathrm{pk}_y}(M+r) \cdot \mathrm{Enc}_{\mathrm{pk}_y}(-r)$
Output: $\mathrm{Enc}_{\mathrm{pk}_y}(M)$

Algorithm 2: SCT protocol.

Input: Alice: $\mathrm{Enc}_{\mathrm{pk}_x}(M_1)$, $\mathrm{Enc}_{\mathrm{pk}_x}(M_2)$
Bob: sk_x.
Begin: Alice:
(a) Pick random numbers $r_1, r_2 \in \mathbb{Z}_N$
(b) Compute $\mathrm{Enc}_{\mathrm{pk}_x}(M_1+r_1) \longleftarrow \mathrm{Enc}_{\mathrm{pk}_x}(M_1) \cdot \mathrm{Enc}_{\mathrm{pk}_x}(r_1)$
(c) Compute $\mathrm{Enc}_{\mathrm{pk}_x}(M_2+r_2) \longleftarrow \mathrm{Enc}_{\mathrm{pk}_x}(M_2) \cdot \mathrm{Enc}_{\mathrm{pk}_x}(r_2)$
(d) Send $\mathrm{Enc}_{\mathrm{pk}_x}(M_1+r_1)$, $\mathrm{Enc}_{\mathrm{pk}_x}(M_2+r_2)$ to Bob
Bob:
(a) Decrypt $(M_1+r_1) \longleftarrow \mathrm{Dec}_{\mathrm{sk}_x}(\mathrm{Enc}_{\mathrm{pk}_x}(M_1+r_1))$
(b) Decrypt $(M_2+r_2) \longleftarrow \mathrm{Dec}_{\mathrm{sk}_x}(\mathrm{Enc}_{\mathrm{pk}_x}(M_1+r_1))$
(c) Compute $\Lambda = (M_1+r_1)(M_2+r_2)$
(d) Compute $\mathrm{Enc}_{\mathrm{pk}_x}(\Lambda)$
(e) Send $\mathrm{Enc}_{\mathrm{pk}_x}(\Lambda)$ to Alice
Alice:
(a) Compute $\mathrm{Enc}_{\mathrm{pk}_x}(M_1 \cdot M_2) = \mathrm{Enc}_{\mathrm{pk}_x}(M_1)^{N-r_2}$
$\mathrm{Enc}_{\mathrm{pk}_x}(M_2)^{N-r_1} \cdot \mathrm{Enc}_{\mathrm{pk}_x}(r_1 r_2)^{N-1} \cdot \mathrm{Enc}_{\mathrm{pk}_x}(\Lambda)$
Output: $\mathrm{Enc}_{\mathrm{pk}_x}(M_1 \cdot M_2)$

Algorithm 3: SM protocol.

4.1.4. Secure Distance Comparison Protocol. Secure distance comparison (SDC) protocol is to determine the shorter distance between two output distances from SDM protocol. Taking as the input two distances, i.e., $(\mathrm{Enc}_{\mathrm{pk}_x}(\Omega_{i,a}), |c_a|)$ and $(\mathrm{Enc}_{\mathrm{pk}_x}(\Omega_{i,b}, |c_b|))$, Alice interacts with Bob to obtain the shorter one. As in [19], the difference between two differences can be expressed as

$$\begin{aligned} &\mathrm{Enc}_{\mathrm{pk}_x}\left(\mathrm{Dist}\left(\vec{d}_i, \vec{\mu}_a\right)\right) \cdot \mathrm{Enc}_{\mathrm{pk}_x}\left(Dist\left(\vec{d}_i, \vec{\mu}_b\right)\right)^{N-1} \\ &\quad = \mathrm{Enc}_{\mathrm{pk}_x}\left(\mathrm{Dist}\left(\vec{d}_i, \vec{\mu}_a\right) - \mathrm{Dist}\left(\vec{d}_i, \vec{\mu}_b\right)\right) \\ &\quad = \mathrm{Enc}_{\mathrm{pk}_x}\left(\Omega/|c_a|^2 - t\Omega_{i,b}/|c_b|^2\right) \\ &\quad = \mathrm{Enc}_{\mathrm{pk}_x}\left(\frac{|c_b|^2\Omega_{i,a} - |c_a|^2\Omega_{i,b}}{|c_a|^2|c_b|^2}\right). \end{aligned} \tag{12}$$

Since we only need to know whether $((|c_b|^2\Omega_{i,a} - |c_a|^2\Omega_{i,b})/(|c_a|^2|c_b|^2)) > 0$ or not, it is equal to judge whether $|c_b|^2\Omega_{i,a} - |c_a|^2\Omega_{i,b} > 0$ or not. This means, the comparison can be related to

$$\mathrm{Enc}_{\mathrm{pk}_x}\left(|c_b|^2\Omega_{i,a} - |c_a|^2\Omega_{i,b}\right). \tag{13}$$

Let β be the maximum size of messages. We have $M \in [-2^\beta + 1, 2^\beta - 1]$, which means $M \bmod N \in [1, 2^\beta - 1]$ if $M > 0$ and $M \bmod N \in [N - 2^\beta + 1, N - 1]$. Let η be the threshold for sign judgement chosen from $[2^\beta - 1, N + 2^\beta - 1]$. To prevent Bob from obtaining distance-related information, Alice blinds the message with a random $r \in [1, \min\{N - \eta, (N - \phi N)/2^{\beta-1}\}]$ with $\phi \in \mathbb{Z}$ and satisfying

$$\begin{cases} (2^\beta - 1) \cdot r \bmod N < \eta \\ (N - 1) \cdot r \bmod N > \eta \\ (N + 1 - 2^\beta) \cdot r \bmod N > \eta \end{cases}. \tag{14}$$

We illustrate the detailed realization in Algorithm 5. In the process, Bob cannot obtain $\Omega_{i,a}, \Omega_{i,b}$.

Input: $\mathrm{Enc}_{\mathrm{pk}_x}(\overrightarrow{d}_i), \mathrm{Enc}_{\mathrm{pk}_x}(\overrightarrow{s_j}), |c_j|$
Begin: $\mathrm{Enc}_{\mathrm{pk}_x}(\Omega_{i,j}) = 0$
for $\alpha = 1$ to m
1. Run SM protocol on $\mathrm{Enc}_{\mathrm{pk}_x}(d_{i,\alpha})$ and $\mathrm{Enc}_{\mathrm{pk}_x}(|c_j|)$ to obtain $\Gamma = \mathrm{Enc}_{\mathrm{pk}_x}(|c_j| \cdot d_{i,\alpha})$
2. Compute $\mathrm{Enc}_{\mathrm{pk}_x}(\Omega_{i,j}) = (\Gamma \cdot \mathrm{Enc}_{\mathrm{pk}_x}(s_{j,\alpha})^{N-1})^2 + \mathrm{Enc}_{\mathrm{pk}_x}(\Omega_{i,j})$

end for
Output: $(\mathrm{Enc}_{\mathrm{pk}_x}(\Omega_{i,j}), |c_j|)$

Algorithm 4: SDM protocol.

4.1.5. Secure Minimum Distance Measurement Protocol. Finally, we define the secure minimum distance measurement (SMDM) protocol as Algorithm 6 to choose the shortest one among given distances.

4.2. Our Scheme. At the beginning, the four entities in the system, i.e., data owners DOs, query client QC, cloud computing service CCS, and key management server KMS, setup the system by running the algorithms, Setup, KeyGen, and AKeyGen. DOs then run Enc and AEnc on their data records and upload to CCS separately. CCS decrypts the received ciphertexts using ADec. After receiving the clustering request from QC, CCS interacts with KMS to transform the ciphertexts encrypted under different public keys to ciphertexts encrypted under the same public key. Subsequently, CCS performs the clustering computation. Finally, CCS interacts with KMS to transfer the clustering result to QC. It is worth noting that the defined protocols are invoked through the process.

4.2.1. System Setup. As the setting in the system model (see Section 4.1), we have n data owners $\{\mathrm{DO}_i\}_{1\le i\le n}$, cloud computing servers (CCS), key management server (KMS), and query client (QC). Before running the protocols, related entities in the system model generate their keys as follows:

(1) Taking as the input a security parameter λ, KMS runs the setup algorithm $\mathrm{Setup}(\lambda)$ of the BCP homomorphic cryptosystem and generate the public parameter pp and master secret key msk, where *msk* is kept secret

(2) Each data owner DO_i runs KeyGen (pp) to generate its own public/secret key pair $(\mathrm{pk}_i, \mathrm{sk}_i)$, $1\le i\le n$

(3) Each DO_i consults with CCS a symmetric key ask_i through Diffie–Hellman key exchange protocol or other methods for $1\le i\le n$

(4) CCS runs the key generation algorithm KeyGen (pp) to generate its public/secret key pair as $(\mathrm{pk}_c, \mathrm{sk}_c)$

(5) QC runs KeyGen (pp) to generate its own public/secret key pair $(\mathrm{pk}_q, \mathrm{sk}_q)$

4.2.2. Data Uploading. Following the setting in Section 4.1, assume that each data owner DO_i has a dataset D_i which contains n_i data records, and each record has m attributes, and DO_i encrypts D_i with BCP cryptosystem first and then AES encryption, $1\le i\le n$. Finally, DO_i sends the output to CCS.

(1) DO_i then runs the encryption algorithm on each record $\overrightarrow{d_j^i} = (d_{j,1}^i, d_{j,2}^i, \ldots, d_{j,m}^i)$, $j \in [1, n_i]$ and obtains the encrypted result as

$$\left\{\mathrm{Enc}_{\mathrm{pk}_i}\left(\overrightarrow{d}_j^i\right)\right\}_{1\le j\le n_i}. \tag{15}$$

(2) To prevent the privacy disclosure from KMS, data owners double-encrypt the output ciphertext with AES encryption. Each DO_i computes

$$\left\{\mathrm{aEnc}_{\mathrm{ask}_i}\left(\mathrm{Enc}_{\mathrm{pk}_i}\left(\overrightarrow{d}_j^i\right)\right)\right\}_{1\le j\le n_i}, \tag{16}$$

and sends the output results to CCS.

(3) After receiving $\left\{\mathrm{aEnc}_{\mathrm{ask}_i}(\mathrm{Enc}_{\mathrm{pk}_i}(\overrightarrow{d}_j^i))\right\}_{1\le j\le n_i}$ from DO_i, CCS runs the decryption algorithm aDEC with the consulted symmetric key ask_i on each ciphertext to obtain

$$\begin{aligned}&\left\{\mathrm{aDEC}_{\mathrm{ask}}\left(\mathrm{aEnc}_{\mathrm{ask}}\left(\mathrm{Enc}_{\mathrm{pk}_i}\left(\overrightarrow{d}_j^i\right)\right)\right)\right\}_{1\le i\le n_i}\\ &\quad=\left\{\mathrm{Enc}_{\mathrm{pk}_i}\left(\overrightarrow{d}_j^i\right)\right\}_{1\le i\le n_i}.\end{aligned} \tag{17}$$

In our setting for data uploading, each data owner DO_i sends their double-encrypted ciphertext to CCS such that the KMS cannot obtain the original message of the data owner although the KMS has the master secret key *msk* which can be used to decrypt the ciphertext encrypted under the BCP homomorphic cryptosystem.

4.2.3. Ciphertext Transformation. This phase is to transfer "multiuser" to "single-user" by re-encrypting the ciphertext encrypted under the public key of pk_i to the ciphertext encrypted under pk_c, $1\le i\le n$.

(1) QC sends a clustering request to CCS.

(2) For a ciphertext $\mathrm{Enc}_{\mathrm{pk}_i}(\overrightarrow{d}_j^i)$ from DO_i, CCS interacts with KMS to run the SCT protocol by setting $\mathrm{pk}_x = \mathrm{pk}_i$, $\mathrm{pk}_y = \mathrm{pk}_c$, $\mathrm{Enc}_{\mathrm{pk}_x}(M) = \mathrm{Enc}_{\mathrm{pk}_i}(\overrightarrow{d}_j^i)$, msk = msk. Finally, CCS obtains $\mathrm{Enc}_{\mathrm{pk}_c}(\overrightarrow{d}_j^i)$.

Input: Alice: $(\mathrm{Enc}_{\mathrm{pk}_x}(\Omega_{i,a}), |c_a|), (\mathrm{Enc}_{\mathrm{pk}_x}(\Omega_{i,b}, |c_b|)), \mathrm{sk}_y$
Bob: $\mathrm{sk}_x, \mathrm{pk}_y$
Begin: Alice interacts with Bob to compute:
(a) $\mathrm{Enc}_{\mathrm{pk}_x}(\Omega'_{i,a}) \longleftarrow SM(\mathrm{Enc}_{pk_x}(\Omega_{i,a}), \mathrm{Enc}_{pk_x}(|_b|^2))$
(b) $\mathrm{Enc}_{\mathrm{pk}_x}(\Omega'_{i,b}) \longleftarrow SM(\mathrm{Enc}_{pk_x}(\Omega_{i,b}), \mathrm{Enc}_{pk_x}(|_a|^2))$
Alice:
(a) Compute $\mathrm{Enc}_{\mathrm{pk}_x}(\Omega'_{i,b}) = (\mathrm{Enc}_{\mathrm{pk}_x}(\Omega'_{i,b}))^{N-1}$
(b) Compute $\mathrm{Enc}_{\mathrm{pk}_x}(\Delta_{a,b}) = \mathrm{Enc}_{\mathrm{pk}_x}(\Omega'_{i,a}) \cdot \mathrm{Enc}_{\mathrm{pk}_x}(\Omega''_{i,b})$
(c) Pick a random number $r \in \mathbb{Z}_N$
Alice interacts with Bob to compute:
(a) $\mathrm{Enc}_{\mathrm{pk}_x}(r\Delta_{a,b}) \longleftarrow SM(\mathrm{Enc}_{\mathrm{pk}_x}(r), \mathrm{Enc}_{\mathrm{pk}_x}(\Delta_{a,b}))$
Alice:
(a) Send $\mathrm{Enc}_{\mathrm{pk}_x}(r\Delta_{a,b})$ to Bob
Bob:
(a) Decrypt $r\Delta_{a,b} \longleftarrow \mathrm{Dec}(\mathrm{Enc}_{\mathrm{pk}_x}(r\Delta_{a,b}))$
(b) If $r\Delta_{a,b} > \eta$, $sn \longleftarrow \mathrm{Enc}_{\mathrm{pk}_y}(1)$; otherwise, randomly choose $r' \in \mathbb{Z}_N$ satisfying $r \neq 1$, $sn \longleftarrow \mathrm{Enc}_{\mathrm{pk}_y}(r')$
(c) Send sn to Alice
Alice:
(a) If $\mathrm{Dec}_{\mathrm{sk}_y}(sn) = 1$, let $\mathrm{Enc}_{\mathrm{pk}_x}(\Omega_{i,\min}) = \mathrm{Enc}_{\mathrm{pk}_x}(\Omega_{i,a})$, $|c_{\min}| = |c_a|$; Otherwise, we have $\mathrm{Dec}_{\mathrm{sk}_y}(sn) \neq 1$, let $\mathrm{Enc}_{\mathrm{pk}_x}(\Omega_{i,\min}) = \mathrm{Enc}_{\mathrm{pk}_x}(\Omega_{i,b})$, $|c_{i,\min}| = |c_b|$
Output: $(\mathrm{Enc}_{\mathrm{pk}_x}(\Omega_{i,\min}), |c_{i,\min}|)$

Algorithm 5: SDC protocol..

Input: $\mathrm{Enc}_{\mathrm{pk}_x}(\vec{d}_i), \mathrm{Enc}_{\mathrm{pk}_x}(\vec{\mu}_1), \mathrm{Enc}_{\mathrm{pk}_x}(\vec{\mu}_2), \ldots, \mathrm{Enc}_{\mathrm{pk}_x}(\vec{\mu}_k)$
Begin: **for** $\alpha = 1$ to k
(a) Run $SDM(\mathrm{Enc}_{\mathrm{pk}_x}(\vec{d}_i), \mathrm{Enc}_{\mathrm{pk}_x}(\vec{\mu}_\alpha))$ and obtain the output $(\mathrm{Enc}_{\mathrm{pk}_x}(\Omega_{i,\alpha}), |c_\alpha|)$
end for
Let $(\mathrm{Enc}_{\mathrm{pk}_x}(\Omega_{i,\min}), |c_{i,\min}|) = (\mathrm{Enc}_{\mathrm{pk}_x}(\Omega_{i,1}), |c_1|)$
for $\alpha = 2$ to k
(a) Run $SDC((\mathrm{Enc}_{\mathrm{pk}_x}(\Omega_{i,\min}), |c_{i,\min}|), (\mathrm{Enc}_{\mathrm{pk}_x}(\Omega_{i,2}), |c_2|))$ and obtain the output $((\mathrm{Enc}_{\mathrm{pk}_x}(\Omega_{i,\min}), |c_{i,\min}|))$
(b) Set $(\mathrm{Enc}_{\mathrm{pk}_x}(\Omega_{i,\min}), |c_{i,\min}|) := (\mathrm{Enc}_{\mathrm{pk}_x}(\Omega_{i,\min}), |c_{i,\min}|)$
end for
Output: $(\mathrm{Enc}_{\mathrm{pk}_x}(\Omega_{i,\min}), |c_{i,\min}|)$

Algorithm 6: SMDM protocol.

(3) By performing the SCT protocol on all the ciphertexts received from $\{\mathrm{DO}_i\}_{1\le i\le n}$, CCS finally obtains

$$\left\{\mathrm{Enc}_{\mathrm{pk}_c}\left(\vec{d}^{\,i}_j\right)\right\}_{1\le i\le n, 1\le j\le n_i}. \tag{18}$$

Let $n = n_1 + n_2 + \cdots + n_l$, and denote these n ciphertexts as

$$\left\{\mathrm{Enc}_{\mathrm{pk}_c}\left(\vec{d}_i\right)\right\}_{1\le i\le n}. \tag{19}$$

For simplicity, we denote $\mathrm{Enc}_{\mathrm{pk}_c}(\vec{d}_i)$ as $\mathrm{Enc}(\vec{d}_i)$ in the following.

It is worth noting that the final ciphertexts are unknown to the KMS since they are blinded in the SCT protocol.

4.2.4. Clustering Computation. In this phase, CCS computes the clustering results with k randomly chosen cluster centers $\mathrm{Enc}(\vec{\mu}_1), \mathrm{Enc}(\vec{\mu}_2), \ldots, \mathrm{Enc}(\vec{\mu}_k)$ from $\left\{\mathrm{Enc}_{pk_c}(\vec{d}_i)\right\}_{1\le i\le n}$.

Let $\mathrm{Enc}(\vec{\mu}_i) = \mathrm{Enc}(\vec{s}_j) = (\mathrm{Enc}(s_{j,1}), \mathrm{Enc}(s_{j,2}), \ldots, \mathrm{Enc}(s_{j,m}))$ and $|c_i| = 1$. CCS also outputs a matrix $V_{n\times k}$ which refers to the location in k clusters of n records, where $V_{i,j} = 1$ means $\vec{d}_i$ is allocated to j-th cluster. In addition, there is a maximum iteration time $\phi_{\max}$. Let $\phi = 0$.

(1) For a data record $\mathrm{Enc}(\vec{d}_i)$, CCS runs the SMDM protocol on it and k-cluster centers with the setting $\mathrm{pk}_x = \mathrm{pk}_c$. Finally, CCS obtains the output

$$\left(\mathrm{Enc}\left(\Omega_{i,\min}\right), \left|c_{i,\min}\right|\right), \tag{20}$$

where $c_{i,\min} = c_\alpha$. Let $V_{i,j} = 0$ for $j \neq \alpha$.

(2) For each data record $\mathrm{Enc}(\vec{d}_i)$ where $\vec{d}_i \neq \vec{\mu}_j$, $1 \le i \le n$, $1 \le j \le k$, CCS runs step 1 and obtains

$$\left(\mathrm{Enc}\left(\Omega_{i,\min}\right), \left|c_{i,\min}\right|\right), \quad 1 \le i \le n, \tag{21}$$

and the matrix $V_{n\times k}$.

(3) With the matrix $V_{n\times k}$ and data records $\mathrm{Enc}(\overrightarrow{d}_i) = (\mathrm{Enc}(d_{i,1}, d_{i,2}, \ldots, d_{i,m}))$, if $V_{i,j} = 1$, CCS updates $|c_j| = |c_j| + 1$ and $\mathrm{Enc}(s_{j,\alpha})$ as

$$\mathrm{Enc}\left(d_{i,\alpha}\right) \cdot \mathrm{Enc}\left(s_{j,\alpha}\right) = \mathrm{Enc}\left(d_{i,\alpha} + s_{j,\alpha}\right), \tag{22}$$

for $1 \le \alpha \le m$. Finally, CCS obtains new $|c_j|$ and $\mathrm{Enc}(\overrightarrow{s}_j) = (\mathrm{Enc}(s_{j,1}), \mathrm{Enc}(s_{j,2}), \ldots, \mathrm{Enc}(s_{j,m}))$ for $1 \le j \le k$. Let $\phi = \phi + 1$.

(4) If $\phi < \phi_{\max}$ and the output matrix $V_{n\times k}$ is different from that in the last iteration, CCS starts a new iteration by running steps (1), (2), and (3). Otherwise, CCS outputs the final

$$\left(\mathrm{Enc}\left(\overrightarrow{s}_j\right), \left|c_j\right|\right)_{1 \le j \le k}. \tag{23}$$

4.2.5. Result Retrieval

(1) CCS interacts with KMS to run the SCT protocol on $\left\{\mathrm{Enc}_{\mathrm{pk}_c}(\overrightarrow{s}_j)\right\}_{1 \le j \le k}$ with the setting $\mathrm{pk}_x = \mathrm{pk}_c$, $\mathrm{pk}_y = \mathrm{pk}_q$, $\mathrm{Enc}_{\mathrm{pk}_x}(M) = \mathrm{Enc}_{\mathrm{pk}_c}(\overrightarrow{s}_j)$. CCS obtains

$$\left\{\mathrm{Enc}_{\mathrm{pk}_q}\left(\overrightarrow{s}_j\right)\right\}_{1 \le j \le k} \tag{24}$$

and sends it and $V_{n\times k}$ to QC.

(2) QC decrypts the received $\mathrm{Enc}_{\mathrm{pk}_q}(\overrightarrow{s}_j)$ with its secret key sk_q by computing

$$\left\{\mathrm{Dec}_{\mathrm{sk}_q}\left(\mathrm{Enc}_{\mathrm{pk}_q}\left(\overrightarrow{s}_j\right)\right) = \overrightarrow{s}_j\right\}_{1 \le j \le k}. \tag{25}$$

QC then computes the cluster centers as

$$\left\{\frac{\overrightarrow{s}_j}{\left|c_j\right|}\right\}_{1 \le j \le k}, \tag{26}$$

where $|c_j| = \sum_{i=1}^{n} V_{i,j}$.

5. Security and Performance Analysis

5.1. Security Analysis. As shown in the proposed scheme (see Section 4.2), our protocol is realized by invoking the BCP homomorphic cryptosystem, AES encryption, and the defined protocols. Upon that, the former two cryptosystems are semantic secure, and we give the security proof of the defined protocols as follows. We take the SM protocol's security proof under "Real-vs.-Ideal" framework as an example. Other protocols' security proofs are in a similar manner and we omit here.

Theorem 1. *SM protocol is secure.*

Proof. SM protocol relates to two semihonest parties, namely, Alice and Bob. Therefore, we consider both securities of SM protocol against semihonest attacker Alice $\mathscr{A}_A$ and semihonest attacker Bob $\mathscr{A}_B$. In the protocol, Alice takes as the input pk_x, $\mathrm{Enc}_{\mathrm{pk}_x}(M_1)$, $\mathrm{Enc}_{\mathrm{pk}_x}(M_2)$ and Bob takes as the input the corresponding secret key sk_x of public key pk_x.

(i) Security against $\mathscr{A}_A$: In the SM protocol, the real-world view of the attacker $\mathbb{Z}_A$ includes the input pk_x, $\mathrm{Enc}_{\mathrm{pk}_x}(M_1)$, $\mathrm{Enc}_{\mathrm{pk}_x}(M_2)$, random numbers r_1, r_2, $\mathrm{Enc}_{\mathrm{pk}_x}(\Lambda)$, and the output $\mathrm{Enc}_{\mathrm{pk}_x}(M_1 \cdot M_2)$, where $\Lambda = (M_1 + r_1)(M_2 + r_2)$. $\mathscr{A}_A$ tries to obtain useful information about the underlying messages, i.e., M_1, M_2, $(M_1 + r_1)(M_2 + r_2), M_1 \cdot M_2$ that are encrypted under pk_x. Because of the semantic security of the used BCP homomorphic cryptosystem, we have that $\mathscr{A}_A$ cannot extract any information of underlying messages except the bit length without sk_x. Therefore, we can construct a simulator $\mathscr{S}_A$ in the ideal world by using ciphertexts of random chosen messages. It will be computationally hard for $\mathscr{A}_A$ to distinguish the ideal world with real world because of the semantic security of the BCP homomorphic cryptosystem. We have

$$\mathrm{Ideal}_{\mathscr{S}_A, \mathscr{A}_A} \overset{c}{\approx} \mathrm{Real}_{\mathrm{SM}, \mathscr{A}_A}, \tag{27}$$

where $\overset{c}{\approx}$ means computationally indistinguishable.

(ii) Security against $\mathscr{A}_B$: In the protocol, $\mathscr{A}_B$ takes as the input the secret key sk_x of pk_x and $\mathrm{Enc}_{\mathrm{pk}_x}(M_1 + r_1)$, $\mathrm{Enc}_{\mathrm{pk}_x}(M_2 + r_2)$. With sk_x, $\mathscr{A}_B$ can decrypt the ciphertexts and obtain the underlying messages $M_1 + r_1, M_2 + r_2$. However, since r_1, r_2 are randomly chosen by Alice, they are random numbers in the point of view of $\mathscr{A}_B$. We can then construct a simulator $\mathscr{S}_B$ in the ideal world by using ciphertexts of random chosen messages, and it will be computationally hard for $\mathscr{A}_B$ to distinguish the ideal world with the real world. We have

$$\mathrm{Ideal}_{\mathscr{S}_B, \mathscr{A}_B} \overset{c}{\approx} \mathrm{Real}_{\mathrm{SM}, \mathscr{A}_B}. \tag{28}$$

This completes the proof of Theorem 1.

Next, we prove that our protocol is secure by taking the process of data uploading as an example. □

Theorem 2. *The data uploading process is secure.*

Proof. In the data uploading process, data owners (DOs) double-encrypt their data records with pk and ask using the BCP homomorphic cryptosystem and AES encryption separately. They then send the encrypted result to CCS who has ask but does not have the corresponding secret key sk of pk. Because of the semantic security of the BCP homomorphic cryptosystem, it is secure against semihonest CCS. Although KMS can extract the underlying messages of ciphertexts encrypted using the BCP homomorphic cryptosystem, it is also computationally hard for a semihonest KMS to obtain any information of data records with the semantic security of AES encryption. Furthermore, CCS and KMS are supposed not to collude in our scheme such that the data

Table 2: The summary of schemes.

Scheme	S/AS	Multiple data owners	Multiple keys	Cipher comparison	Security	Multidimensional data
[22]	AS	×	×	×	×	√
[23]	S	×	×	√	√	√
[26]	AS	√	×	√	√	√
[30]	AS	√	√	√	×	√
[34]	AS	√	√	√	×	√
[19]	AS	√	√	√	×	√
Ours	*AS*	√	√	√	√	√

S/AS: symmetric/asymmetric.

uploading process is secure against semihonest CCS and KMS separately. This completes the proof of Theorem 2.

It is worth noting that the security of our construction is protected by the semantic security of the BCP homomorphic cryptosystem, AES encryption, and blinding with random numbers, which prevents the adversaries from obtaining any useful information from the received ciphertexts. □

5.2. Performance Analysis. In our construction, we use the BCP homomorphic cryptosystem and AES encryption to encrypt data owners' data records to prevent the information disclosure to KMS. Compared with the underlying scheme [19] which utilizes Youn's homomorphic encryption scheme [33], our scheme therefore increases the computation cost between DOs and CCS along with the increased security.

In particular, each data owner additionally needs to interact with CCS to consult a symmetric key of AES encryption in the system setup phase. Except this, since BCP encryption has additive homomorphic property instead multiplication in Youn's encryption scheme [33], we give a secure multiplication protocol SM instead of secure addition SA in [19]. This leads to different invocations in other defined protocols, which result in more computation cost.

With the sacrifice on the computation cost, our scheme achieves semantic security that adversaries cannot obtain any useful information about underlying data records, while AS can extract M_1/M_2 in SA protocol of [19]. Furthermore, in our scheme, KMS cannot extract the underlying data records of data owners, while KMS can realize this with its master secret key in [19].

Finally, we compare our scheme with the existing outsourced k-means clustering schemes [19, 22, 23, 26, 30, 34] in Table 2 on six aspects, i.e., whether the scheme is based on symmetric or asymmetric cryptosystem, whether it supports or achieves multiple data owners and multiple keys, ciphertext comparison, security, and multidimensional data. As shown in Table 2, our scheme achieves all the listed functionalities under the asymmetric cryptosystem.

6. Conclusions

This paper proposed a highly secure privacy-preserving outsourced k-means clustering scheme on the encrypted datasets under multiple keys. We utilized BCP homomorphic encryption and AES encryption to double-encrypt the data records in the database to protect the security against semihonest cloud computing server and key management server. Furthermore, we constructed five protocols, i.e., secure ciphertext transformation (SCT), secure multiplication (SM), secure distance measurement (SDM), secure distance comparison (SDC), and secure minimum distance measurement (SMDM), as the base of our scheme. In particular, SM protocol is built to achieve the homomorphic multiplicative property using BCP encryption. Finally, we proposed our scheme by invoking the defined protocols thoroughly. The given security and performance analysis shows that our scheme is comparable with the existing outsourced k-means clustering scheme on security and functionality.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Key R&D Program of China under Grant no. 2017YFB0802000, the National Natural Science Foundation of China under Grant nos. 61572390 and U1736111, the National Cryptography Development Fund under Grant no. MMJJ20180111, the Plan For Scientific Innovation Talent of Henan Province under Grant no. 184100510012, the Program for Science & Technology Innovation Talents in Universities of Henan Province under Grant no. 18HASTIT022, and the Innovation Scientists and Technicians Troop Construction Projects of Henan Province.

References

[1] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323, 1999.

[2] A. K. Jain, "Data clustering: 50 years beyond k-means," *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651–666, 2010.

[3] J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, and W. Lou, "Fuzzy keyword search over encrypted data in cloud computing," in *Proceedings of the INFOCOM*, pp. 441–445, IEEE, San Diego, CA, USA, March 2010.

[4] V. Goyal, "Reducing trust in the PKG in identity based cryptosystems," in *Proceedings of the CRYPTO 2007, 27th*

Annual International Cryptology Conference, Lecture Notes in Computer Science, vol. 4622, pp. 430–447, Springer, Santa Barbara, CA, USA, August 2007.

[5] D. A. Davis, N. V. Chawla, N. Blumm, N. A. Christakis, and A. Barabási, "Predicting individual disease risk based on medical history," in *Proceeding of the 17th ACM Conference on Information and Knowledge Mining*, pp. 769–778, ACM, Napa Valley, CA, USA, October 2008.

[6] J. Li, H. Yan, and Y. Zhang, "Certificateless public integrity checking of group shared data on cloud storage," *IEEE Transactions on Services Computing*, pp. 1–10, 2018.

[7] H. Yan, J. Li, and Y. Zhang, "Remote data checking with a designated verifier in cloud storage," *IEEE Systems Journal*, pp. 1–10, 2019.

[8] J. Li, H. Yan, and Y. Zhang, "Efficient identity-based provable multi-copy data possession in multi-cloud storage," *IEEE Transactions on Cloud Computing*, p. 1, 2019.

[9] G. Wu, Y. Mu, W. Susilo, F. Guo, and F. Zhang, "Threshold privacy-preserving cloud auditing with multiple uploaders," *International Journal of Information Security*, vol. 18, no. 3, pp. 321–331, 2019.

[10] G. Wu, Y. Mu, W. Susilo, F. Guo, and F. Zhang, "Privacy-preserving certificateless cloud auditing with multiple users," *Wireless Personal Communications*, vol. 106, no. 3, pp. 1161–1182, 2019.

[11] G. Wu, Y. Mu, W. Susilo, and F. Guo, "Privacy-preserving cloud auditing with multiple uploaders," *Information Security Practice and Experience*, vol. 10060, pp. 224–237, 2016.

[12] A. Blum, C. Dwork, F. McSherry, and K. Nissim, "Practical privacy: the sulq framework," in *Proceedings of the ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, ACM*, pp. 128–138, Baltimore, MD, USA, June 2005.

[13] Q. Yu, Y. Luo, C. Chen, and X. Ding, "Outlier-eliminated k-means clustering algorithm based on differential privacy preservation," *Applied Intelligence*, vol. 45, no. 4, pp. 1179–1191, 2016.

[14] J. Ren, J. Xiong, Z. Yao, R. Ma, and M. Lin, "Dplk-means: a novel differential privacy k-means mechanism," in *Proceedings of the 2017 IEEE Second International Conference on Data Science in Cyberspace (DSC)*, pp. 133–139, Shenzhen, China, June 2017.

[15] T. Shang, Z. Zhao, Z. Guan, and J. Liu, "A DP canopy k-means algorithm for privacy preservation of hadoop platform," in *Proceedings of the CSS 2017, Lecture Notes in Computer Science*, vol. 10581, pp. 189–198, Springer, Xi'an, China, October 2017.

[16] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Proceedings of the EUROCRYPT'99, Lecture Notes in Computer Science*, vol. 1592, pp. 223–238, Springer, Prague, Czech Republic, May 1999.

[17] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 26, no. 1, pp. 96–99, 1983.

[18] M. Kantarcioglu and C. Clifton, "Privacy-preserving distributed mining of association rules on horizontally partitioned data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 9, pp. 1026–1037, 2004.

[19] H. Rong, H. Wang, J. Liu, J. Hao, and M. Xian, ""Outsourced k-means clustering over encrypted data under multiple keys in spark framework," in *Proceedings of the SecureComm 2017, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, vol. 238, pp. 67–87, Springer, Niagara Falls, ON, Canada, October 2017.

[20] Q. Zhang, L. T. Yang, Z. Chen, and P. Li, "Pphopcm: privacy-preserving high-order possibilistic c-means algorithm for big data clustering with cloud computing," *IEEE Transactions on Big Data*, 2017.

[21] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "(Leveled) fully homomorphic encryption without bootstrapping," in *Innovations in Theoretical Computer Science*, pp. 309–325, ACM, Cambridge, MA, USA, 2012.

[22] N. Almutairi, F. Coenen, and K. Dures, "K-means clustering using homomorphic encryption and an updatable distance matrix: secure third party data clustering with limited data owner interaction," in *Proceedings of the DaWaK 2017, Lecture Notes in Computer Science*, vol. 10440, pp. 274–285, Springer, Lyon, France, August 2017.

[23] J. Yuan and Y. Tian, "Practical privacy-preserving mapreduce based k-means clustering over large-scale dataset," *IEEE Transactions on Cloud Computing*, vol. 7, no. 2, pp. 568–579, 2019.

[24] O. Regev, "On Lattices, learning with errors, random linear codes, and cryptography," in *ACM Symposium on Theory of Computing*, pp. 84–93, ACM, Baltimore, MD, USA, 2005.

[25] K.-P. Lin, "Privacy-preserving kernel k-means clustering outsourcing with random transformation," *Knowledge and Information Systems*, vol. 49, no. 3, pp. 885–908, 2016.

[26] F. Rao, B. K. Samanthula, E. Bertino, X. Yi, and D. Liu, "Privacy-preserving and outsourced multi-user k-means clustering," in *Proceedings of the CIC 2015*, pp. 80–89, IEEE Computer Society, Hangzhou, China, October 2015.

[27] Y. Liu, Y. Luo, Y. Zhu, Y. Liu, and X. Li, "Secure multi-label data classification in cloud by additionally homomorphic encryption," *Information Sciences*, vol. 468, pp. 89–102, 2018.

[28] Z. Gheid and Y. Challal, "Efficient and privacy-preserving k-means clustering for big data mining," in *Proceedings of the 2016 IEEE Trustcom/BigDataSE/ISPA*, pp. 791–798, IEEE, Tianjin, China, August 2016.

[29] C. Clifton, M. Kantarcioglu, J. Vaidya, X. Lin, and M. Y. Zhu, "Tools for privacy preserving distributed data mining," *ACM Sigkdd Explorations Newsletter*, vol. 4, no. 2, pp. 28–34, 2002.

[30] A. Peter, E. Tews, and S. Katzenbeisser, "Efficiently outsourcing multiparty computation under multiple keys," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 12, pp. 2046–2058, 2013.

[31] Y. Li, Z. L. Jiang, X. Wang, S. Yiu, and J. Fang, "Outsourced privacy-preserving random decision tree algorithm under multiple parties for sensor-cloud integration," in *Proceedings of the ISPEC 2017, Lecture Notes in Computer Science*, vol. 10701, pp. 525–538, Springer, Melbourne, Australia, December 2017.

[32] E. Bresson, D. Catalano, and D. Pointcheval, "A simple public-key cryptosystem with a double trapdoor decryption mechanism and its applications," in *Proceedings of the ASIACRYPT 2003, Lecture Notes in Computer Science*, vol. 2894, pp. 37–54, Springer, Taipei, Taiwan, November 2003.

[33] T. Youn, Y. Park, C. H. Kim, and J. Lim, "An efficient public key cryptosystem with a privacy enhanced double decryption mechanism," in *Proceedings of the SAC 2005*, vol. 3897, pp. 144–158, Springer, Kingston, ON, Canada, August 2005, Lecture Notes in Computer Science.

[34] X. Liu, R. H. Deng, K.-K. R. Choo, and J. Weng, "An efficient privacy-preserving outsourced calculation toolkit with multiple keys," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 11, pp. 2401–2414, 2016.