Hindawi
Security and Communication Networks
Volume 2019, Article ID 9108759, 14 pages
https://doi.org/10.1155/2019/9108759

Research Article

Privacy Protection of Social Networks Based on Classified Attribute Encryption

Lin Zhang[1,2,3] **Li Li**[1] **Eric Medwedeff**[2] **Haiping Huang**[1,3] **Xiong Fu**[1,3] **and Ruchuan Wang**[1,3]

[1]*College of Computer, Nanjing University of Posts and Telecommunications, Nanjing 210003, China*
[2]*Department of Computer Science, San Diego State University, San Diego, CA 92182, USA*
[3]*Jiangsu High Technology Research Key Laboratory for Wireless Sensor Networks, Nanjing 210003, China*

Correspondence should be addressed to Lin Zhang; zhangl@njupt.edu.cn

Received 12 February 2019; Revised 12 April 2019; Accepted 25 August 2019; Published 26 September 2019

Guest Editor: Hua Wang

With the rapid development of social networks, privacy has also attracted attention. Based on this problem, a privacy protection scheme for social networks based on classified attribute encryption (PPSSN) is proposed for the data owner and attribute management server to manage user permissions; the approach reduces data owner overhead and also avoids use of a property management server to limit access user collusion attacks. To balance the privacy and security of data publication, this scheme classifies users and designs access control for different users and different privileges. In addition, this paper also introduces a good friend data cache mechanism to improve and optimize the original scheme to reduce the cost of decryption. The efficiency and system overhead of the proposed scheme are compared and analyzed based on experiments. The experiments show that the proposed scheme improves query efficiency, reduces system cost, and enhances privacy security.

1. Introduction

In this era of rapid information technology and networking, people have greater access to data. With the convenience of communication, social networks have come into being and gradually became popular. At present, the definition of social networking is relatively mature; that is, social networking services are built [1] for social networking applications. Through social networking, we can communicate with friends by sending messages and sharing content. Because of its convenience, ease of operation, and so on, social networks have become more and more common, given the maturity of network technology. At the same time, with the rapid development of social networking, the problem of privacy has also attracted attention. Private data in social networks mainly include user identity information, user login information, user friend information, and data published on the social network platform. The root cause of privacy security in social networks is that the data owner's privacy data are separated from the direct physical control of the data owner when the data are transmitted on the social network platform. As a result, the data may be leaked, and thus users who have not been able to view the permission or even users who are malicious steal information to see the content of the data.

To date, lots of research attention has been given to security and privacy of network technology. Zhang et al. [2] established a theoretical framework for the study of eavesdropper-tolerance capability in a two-hop wireless network, where the cooperative jamming is adopted to ensure security defined by secrecy outage probability (SOP) and opportunistic relaying is adopted to guarantee reliability defined by transmission outage probability. To tackle the identified challenges, Wang et al. [3] summarized high-quality submissions for the special issue, the paper generalized that Lomotey et al. [4] proposed a provenance technique, which leverages the associative rules and lexical chaining methodologies to enable data traceability, as well as the detection of faulty data propagation, through the identification of propagation routes of data and object-to-

object communications. And Chen et al. [5] proposed a set of cryptographic techniques to protect medical IoT with respect to data transmission, storage and access, including a multipath asymmetric encryption fragment transmission mechanism, a distributed symmetric encryption cloud storage scheme, and a dynamic access control scheme. Huang et al. [6] proposed an access control mechanism based on hierarchical attribute-based encryption (ABE) scheme to preserve the confidentiality of collected data in transmission and at rest (i.e., stored at resource constrained IoT devices).

For the above privacy protection, encryption is a common solution. For sensitive data, ciphertext access control can be used to control user decryption privileges by encrypting data. Shamir [7] puts forward the concept of the identity-based system, and Boneh and Franklin [8] put forward the self-identity-based encryption scheme based on this. In the attribute-based scheme proposed by Sahai and Waters [9], the user's qualification certificate is expressed by the attribute set, and the operation is expressed by the formula of these attributes to solve the problem of decryption of the ciphertext by sharing the data. However, one drawback is that their initial structure is limited to a formula made up of thresholds.

Goyal et al. [10] then divided the property attribute-based encryption schemes into two categories: one is an encryption scheme based on the key strategy, and the other is an attribute encryption scheme based on the ciphertext strategy. In the encryption scheme based on the key strategy, the key is associated with the access control strategy, and in the cryptographic scheme based on the ciphertext strategy, the ciphertext is associated with the control strategy, and the key is associated with the attribute set. Bethencourt et al. [11] proposed an attribute encryption technology based on the ciphertext strategy (CP-ABE, ciphertext-polity attribute-based encryption) to associate the user's private key to a set of attributes, and the user ciphertext is associated with the access tree. When the user attribute set satisfies the access tree, the user can decrypt the data. The algorithm contains the decryption rules in the encryption algorithm, which greatly optimizes the frequent key distribution in ciphertext access. Waters [12] proposed a new method of the attribute-based encryption of the ciphertext policy that combines the linear secret-sharing scheme. In the framework, three structures are proposed to realize the specific and noninteractive assumptions of the general access structure of the cipher policy ABE system from the standard model.

Because most of the existing attribute-based schemes are mostly based on pairing-based operations, the size and decryption time of the ciphertext will increase with the expansion of the access scale, requiring huge computing costs. Aiming at this problem, Green et al. [13] put forward a secure outsourcing computing technology that converts most of the decoding work to the proxy cloud server. However, because the identity of the user is not always fixed in the system, any change in user identity means that the attribute will be replaced, revoked, or added, so the scheme has the problem of user flexibility in attribute revocation.

In the property revocable scheme proposed by Yu et al.[14], the agent can delegate the agent's re-encryption and key update to the proxy server by proxy re-encryption. When users access encrypted shared data, the proxy server re-encrypts them. Although the scheme has provided formal security proof for unauthorized users to use a semitrusted proxy server, it does not provide formal security proof for the revoked user.

In Naruse et al. [15] scheme, no user can decrypt ciphertext encrypted by a public key generated by authority. When users download encrypted shared data from cloud servers, the encrypted proxy of the cloud server is re-encrypted, and users can decrypt it. When the cancelled user downloads the encrypted shared data, the cloud server will not re-encrypt it. So, the revoked user cannot decrypt it; however, because the scheme only supports the "and" policy, the access strategy is limited.

In a one-to-many encryption mechanism, users with the same attributes will share the same decryption privileges. It is possible that the malicious users can divulge the decryption key or decryption privileges to others in the form of a decryption black box. To solve this problem, Liu et al. [16, 17] used the black box traceability of system's access to detect whether the user was leaking the decryption device's behavior. However, the problem of how to effectively revoke permissions after malicious users and to design efficient black boxes in a short ciphertext-based system without sacrificing other performance remains unsolved.

In CP-ABE, according to the authorization center, it can be divided into two categories: the data center and authority center. In [18, 19], the idea is that the data owner is the center, and the generation and distribution of the private key is completed by the data owner when the data owner establishes contact with the user. Liang et al. [20] proposed that the authority of the attribute authority is the center, which is responsible for the work of generating and distributing the private key by the authority, and the data owner is only responsible for the encryption of the data. The two ideas are to ensure that only users who satisfy the attribute policy can decrypt the published data. However, the cost of generating the private key is linearly related to the corresponding set of attributes, so the data-owner center leads to bottlenecks in data-owner computing when access to the user is greatly increased; the authority-centered scheme has the effect of reducing the data's main cost; the authority will bring the risk of data leakage at the same time so that the privacy of the network data cannot be guaranteed.

Lv et al. [1] have made new improvements on the basis of attribute-based encryption and designed the attribute-based encryption algorithm with trapdoor to achieve efficient cancellation. However, it is not satisfied with the refine service.

On the basis of [1], in order to adapt to diverse user needs, Lin et al. [21] proposed a privacy protection scheme based on attribute encryption to support more precise services. It provides three modes to perform a simple classification of friends, make different attribute strategies for them, and get different precise ciphertexts from different levels. Although the scheme is capable of resisting social

network friend attacks, it is necessary to encrypt three different operating attributes according to the difference in key information, which increases the amount of computation and increases the cost. Moreover, this study only focuses on location information encryption and is not widely applicable.

In view of the above problems, this paper proposes a social network privacy protection scheme based on attribute encryption (PPSSN) and designs the following improvements:

(1) The access authority of visitors is limited by data owners and attribute management servers, which not only reduces the overhead of the data owner but also handles part of the work by the attribute management server, thus avoiding conspiracy attacks between the attribute management server and the nonprivileged access user.

(2) To tradeoff the usability of users' published data and of information privacy protection, according to the data owner classification of the relationship between all users on a social network, the mechanism of different close relationships accessing data with different degrees of accuracy is adopted, and the access control of different rights is realized.

(3) To reduce the decryption cost of the system, the friends data buffer mechanism is introduced to improve and optimize it.

The organizational structure of this paper is as follows: Section 2 introduces the related concepts; Section 3 describes the overall design of the scheme in detail, including the system model, the algorithm design, and the scheme description; Section 4 proves the security of the scheme; and Section 5 compares the feasibility and efficiency analysis of the case with the experiment.

2. Related Works

2.1. Bilinear Mapping

Definition 1 (bilinear mapping [8]). For the multiplicative groups G_1 and G_2, for which the rank is a large prime number p and g is a generator of G_1. The bilinear map e: $G_1 \times G_1 \longrightarrow G_2$ has the following properties:

Bilinear: for any a, $b \in Zp$, for $g \in G_1$, $e(g_a, g_b) = e(g, g)ab$

Non degeneracy: given $g \in G_1$, make $e(g, g) \neq 1$

Computability: for any $u, v \in G_1$, $e(u, v)$ can be effectively calculated

Definition 2 (access structure [11]). Let us set up $P = \{P_1, P_2, \ldots, P_n\}$ is a collection of all user attributes, and each user-associated attribute set is $A \subseteq P = \{P_1, P_2, \ldots, P_n\}$, then the access structure A is the nonempty subset of the attribute set P. The access structure A represents an attribute judgment condition, and the set of elements in the A is called an authorization set, and the set not made up of elements in the A is called an unauthorized set, and only the authorized user's key can decrypt the corresponding file.

2.2. Access Structure and Access Tree.

The access tree is used in this article to describe the access structure. The access tree T is used to describe an access control strategy, which is a nonempty subset of the complete set of attributes P, T represents an attribute judgment condition, the leaf node of T represents an attribute, and a non-leaf node represents a relational function, in which the relation function is an "and", "or", "n of m" threshold. The set of attributes in T is authorized and those not in T is not authorized. Only the authorized user's key can decrypt the file. In an attribute-based encryption structure, the access control strategy is represented by the access structure tree, which can be decrypted only when the attribute set associated with the user's private key meets the access structure tree [1].

The schematic diagram of the access tree structure is shown in Figure 1.

2.3. Security Hypothesis.

In this scheme, it is assumed that the attribute management system (AMS) and the data owner (DO) server are not fully trusted (honest but curious), that the AMS and DO servers will execute the corresponding program not to disclose the information actively but may try to obtain more information such as access rights and the exact data content of the distribution, resulting in some information being used by malicious users or other external attackers. It is assumed that the additional information obtained by the AMS and DO server is user-related identity information, permission to file, and the content of published file data. In addition, this paper assumes that the information transmission channel between the DO and AMS and between the DO and DO server is secure.

Because this scheme is designed for access control for general sensitive data, not top secret information, the AMS will choose the correct execution program, so the security hypothesis of this article is reasonable.

3. Model and Algorithm Design

3.1. System Model.

The concepts in trust valuation and roles of nodes are as follows:

As shown in Figure 2, the social network privacy protection system (PPSSN) model contains the data owner (DO), the DO sever, the attribute management sever (AMS), the visitors, and the social networking service platform (SNP). Among them, the function of the DO is to classify and generate a friend list according to the intimate degree of the friends, determine the attribute encryption strategy, and generate and distribute the main private key of the data. The DO server encrypts and stores the friends list of DO. When the user requests access to data, it queries the encrypted list and returns the friendship data, which can share the DO overhead, but does not affect the security of the data. After receiving the user's access request, the AMS applies for a friend relationship to the DO, judges the friend's

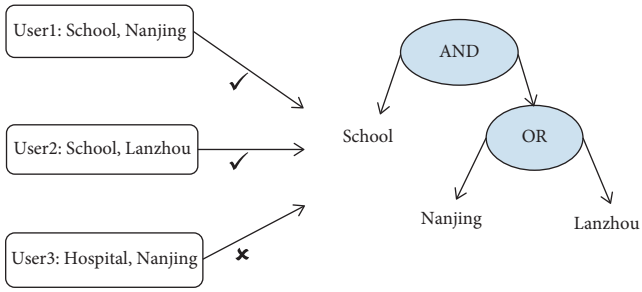

Figure 1: A schematic diagram of the structure of an access tree.

relationship through the data returned by the DO, and distributes the private key to the visitor. The social networking service platform (SNP) is responsible for user data publishing. The visitor V is the user on the social networking platform. When checking the publishing data of the DO, they need the corresponding access authority to access the data on the SNP.

When the visitor requests access to data, it needs to first apply to the DO through authentication of the AMS and DO and next receive the private key to access the SNP data; different identities will have different access rights. Close friends of the DO can get the first encryption parameters and can access the original accurate data of the DO, and the general friends of the DO can only access the published data that are encrypted after one time. According to this, users of the non-DO's friends can only access data that were last posted on the SNP, and illegal users identified by the AMS as social networks cannot access any DO published data.

3.2. Algorithm Design. This section introduces some core algorithm functions in the process of file re-encryption. On the basis of [8], the revocation is improved and set up to support the user level. The main functions are: the Grek function used to generate the re-cryptographic key, the Fre function for the re-encrypted operation of the file, and the CoA function to update the private key of the user.

3.2.1. Grek(): Generates a Re-Cryptographic Key. The main ideas of the algorithm are as follows: first, input the tree's leaf node attribute set, generate a random number, and recalculate the new private key of leaf node, then generate a new key, that is, encryption key, and finally replace the new key and the related parameters.

The basic description of the algorithm is shown in Algorithm 1.

3.2.2. Fre(): File Re-Encryption. The main ideas of the algorithm are as follows: on the basis of the original ciphertext, according to the new key generated by the function Grek (), the structure tree and the original ciphertext, recalculate the ciphertext and update the relevant parameters to realize the re-encryption of the file.

The algorithm flow is shown in Algorithm 2.

3.2.3. CoA(): The Replacement of the Permissions. The main ideas of the algorithm are as follows: according to the corresponding input, the AMS is responsible for revocation of authority and updating the private key of the unrevoked user at the same time, so that revoke the access rights of the unauthorized users and avoid the unauthorized access.

The algorithm flow is shown in the following Algorithm 3.

4. Scheme Design

4.1. Initialization

4.1.1. System Initialization. The DO completes the system's key initialization and sets related parameters.

Setup() $\longrightarrow$ OP, SMK: Define a bilinear mapping e: $G_0 \times G_0 \longrightarrow G_T$, where G_0 is a bilinear group, and it is constructed by P, whose rank is prime, and generator g. Define attribute space $A = \{A_1, A_2, \ldots, A_n\}$, for any property $A_i \in A\,(1 \le i \le n)$ randomly selected $x_{A_i} \in Z_p$ (p and cyclic group), randomly generated $\alpha, \beta \in Z_p$.

Published parameter OP = $\{G_0, g, g^{\beta}, g^{x_{A_i}}, e(g,g)^{\alpha}\}$

Generate system master key SMK = $\{\beta, g^{\alpha}, \{x_{A_i}\}\}$

The DO runs Setup(), generating public parameter OP and system master key SMK.

4.1.2. Friends List Initialization. Set up a list of friends in the DO server. The statistics and display information in the friend list are serial number (SN), access user ID, friend relationship parameter (FR), and verification parameter (VP).

The initial list shows close friends and ordinary friends identified by the DO. If you are close friends, the friend relationship parameter is 1; if you are an ordinary friend, you can get a friend relationship parameter (FR) to 0.

The BLori is defined as a list of original friends. In the BLori, for each record, the friend list has a serial number (SN), a user ID, a friend relationship parameter (FR), and a validation parameter (VP).

Among them, $\mathrm{BL_{ori}} = \{\mathrm{Rl}[1], \mathrm{Rl}[2], \ldots, \mathrm{Rl}[n]\}$, where Rl $[i]$ represents the record of article i in the $\mathrm{BL_{ori}}$ $(1 \le i \le n)$.

4.1.3. Friend List Encryption. Encrypted $\mathrm{BL_{ori}}$ to generate $\mathrm{BL_{en}}$, which is an encrypted friends list.

$\mathrm{BL_{en}} = \{\mathrm{Rl}'[1], \mathrm{Rl}'[2], \ldots, \mathrm{Rl}'[n]\}$, and Rl′ $s[i]$ represents the record of article i in the friend list $\mathrm{BL_{en}}$ $(1 \le i \le n)$; the function of the $\mathrm{BL_{en}}$ list is that the DO server does not understand the correspondence between the encrypted list of friends and the record of the original friends list and prevents the DO server from leaking the friend list information.

Let $\mathrm{BL_{ori}}$ and $\mathrm{BL_{en}}$ be the matrices of $\{0,1\}^{n*1}$, where n is the number of lists recorded and l is the length of the list record (unit bit); that is, an arbitrary record i is a vector as $\{0, 1\}^1 (1 \le i \le n)$, $\mathrm{Rl}[i] = (b_{i1}, b_{i2}, \ldots, \mathrm{b}_{im})$, $\mathrm{Rl}'[i] = (b'_{i1}, b'_{i2}, \ldots, b'_{im})$ $(m = 1/32 + 1)$, Rl$[i]$ and Rl′$[i]$ are divided into 32 bit blocks, $\mathrm{BL_{ori}}$ and $\mathrm{BL_{en}}$ can be expressed as

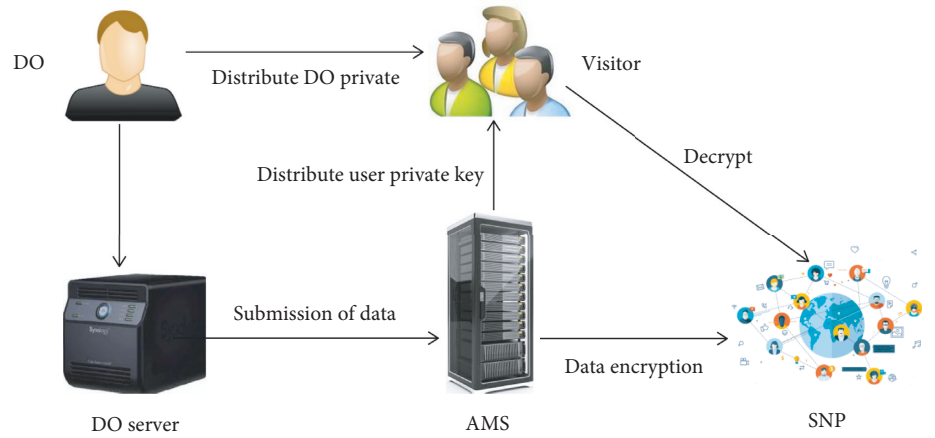

Figure 2: System model.

Input: the set of leaf node attributes of the access tree
Output: new key
//Generate a new key for each leaf node
(1) if ($i \notin$ URL)
exit;
(2) else
(3) take any $s' \in Z$;
(4) for $z \in Z$, generate a new set of leaf node attributes $q'_z(0)$
(5) compute new private key of leaf node;
(6) take any $x'_i \in Z_p$;
(7) calculate the re-encryption key according to the new access tree; //Generate a new key
(8) replace the new key PPK';
(9) replace a new signature;
(10) $\mathrm{OP}'_x = \mathrm{OP}'_{1,x} = \left\{g^{\beta x}, \mathrm{OP}'_{2,x} = g^{1/\beta x}\right\}$ //Compute new public parameters

Algorithm 1: Generates re-encryption key.

Input: access structure tree, new encryption key, ciphertext CT
Output: CT′
(1) //Calculate the encrypted C'_1, C'.
(2) $C'_1 = \mathrm{CT} \cdot e(g, g)^{\alpha s'}$;
(3) $C' = g^{\beta s'}$;
(4) $C_z'^{(1)} = g^{q' z(0)}$,
(5) $C_z'^{(2)} = g_{\mathrm{att}(z)}^{xiq' z(0)}$;
(6) $\mathrm{CT}' = \{T', C'_1, C, C_z'^{(1)}, C_z'^{(2)}\}$;

Algorithm 2: File re-encryption.

Input: the user i, the new private key
(1) if ($i \in$ URL)//Determine it in the user revocation list or not
(2) exit;
(3) else
(4) distribute the updated parameters to the user i;
(5) for each attribute of the user i
(6) calculate the corresponding key;
(7) replace the property private key;
(8) the user updates the private key;
(9) end for

Algorithm 3: Permission replacement algorithm.

$$\mathrm{BL_{ori}} = \begin{matrix} b_{11} & b_{12} & \cdots & b_{13} \\ b_{21} & b_{22} & \cdots & b_{23} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \cdots & b_{nm} \end{matrix}, \quad \mathrm{BL_{en}} = \begin{matrix} b'_{11} & b'_{12} & \cdots & b'_{13} \\ b'_{21} & b'_{22} & \cdots & b'_{23} \\ \vdots & \vdots & & \vdots \\ b'_{n1} & b'_{n2} & \cdots & b'_{nm} \end{matrix}, \tag{1}$$

where $\mathrm{BL_{ori}}$ encryption generates the $\mathrm{BL_{en}}$ process as shown in Figure 3.

The main idea of the encryption process is to first set up a mapping table that records the corresponding relationship between $\mathrm{BL_{ori}}$ and $\mathrm{BL_{en}}$ in DO, select a random number as the seed, and generate a random number sequence; then, the process reads into the data of $\mathrm{BL_{ori}}$, performs a modular operation of the corresponding bit with a pseudorandom sequence, saves the result into the register, rearranges the register according to the mapping table, and finally writes the register data to $\mathrm{BL_{en}}$. To ensure the privacy of data in the encryption process, a random number of encrypted $\mathrm{BL_{ori}}$ and some parameters are generated by DO. Among them, the encryption algorithm is based on the pseudorandom encryption algorithm of [18], and the specific encryption algorithm is shown in the following Algorithm 4.

4.1.4. Validation List V Initialization. A verification list V is built in the DO server, and it is used to record users who have been visited after the DO data release and are judged to be close friends by the DO.

The information shown in the verification list is: serial number (SN) and access to the user ID.

In addition, the information in the initial time list is empty except for the header, and users will update the verification list after the user's access is satisfied. When the user deletes the published content, the corresponding verification list page will be deleted.

Since only close friends who have already visited can join the list, the DO can first query Algorithm 5 when accessing the user's application data access. If it has previously been designated as a close friend, it can jump directly over the friend relationship judgment and go directly to the next step. It can reduce the query amount of subsequent queries and the cost.

4.1.5. Initialize the User Revocation List (URL). A user revocation list (URL) is set up in the AMS, which records all users who will be revoked by the DO. When an attribute is revoked, the user access to the file is revoked by referring to the URL table.

The information in the URL table includes the identification of the user ID and its cancelled files.

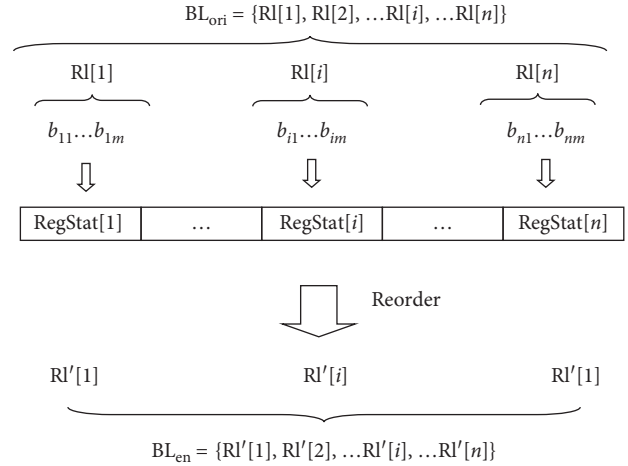

Figure 3: Encryption process.

4.2. DO Data Release. The DO will publish the content to be shared through the SNP platform. Before the data are released on the platform, the source data will be processed accordingly. This scheme describes the encryption processing of single file data, including the encryption of the original data by the DO, the generation and distribution of the private key, and the data release in the social network platform.

4.2.1. The DO Processes the Raw Data to Satisfy the Privacy Difference with Privacy Budget ε. For the stored source data di, the DO is processed first, and only the DO authenticated as a close friend can obtain the relevant parameters and ensure that it has the right to obtain the accurate data issued by the DO.

Step:

(1) Input di;

(2) Add noise to make it satisfy the differential privacy protection;

(3) Output: di′

4.2.2. Attribute-Based Encryption

Step 1: in the DO, define the mapping relationship between the attribute space A and the user, and the corresponding attribute set S is generated according to the user identity information. The KeyGen function is executed to generate the corresponding private key and send it to the user through the security channel.

PPK = KeyGen(S, SMK): The set of attributes associated with the user private key is defined as $S = \{S_1, S_2, \ldots, S_k\}$ (S is a subset of A). For any $S_j \in S$ ($1 \le j \le k$), randomly select $y_{Sj} \in Z_p$, randomly generates gamma $\gamma \in z_p$ and then generates a master key PPK. PPK = $\{D = g^{(\alpha+\gamma)/\beta}, \forall S_j \in S: D_{s_j}^{(1)} = g^{\gamma} g^{X_{ai}} g^{y_{sj}}, D_{s_j}^{(2)} = g^{y_{sj}}\}$.

Input: BL_{ori}
Output: BL_{en}
(1) for ($i = 1$; n; i++)
(2) map[i] = random (1, n)//Return an integer type random number between 0 and 1;
(3) end for ($i = i$; n; i++)
(4) for ($i = i$; n; i++)
(5) RegStat[0] = (rand() << 17) | (rand() << 3) | (rand()); //Generate random number store in the register;
(6) RegStat[0]take as seed, and generate pseudorandom number sequence PRS = (p_1, p_2, ..., p_n);
(7) for ($j = 1$; n; j++)
(8) RegStat[j] = $b_{ji} \oplus p_j$;
(9) RegStat[map[t]] = RegSta[i]; //Rearrangement of registers by mapping table
(10) end for
(11) for ($k = 1$; n; k++)
(12) add RegStat[k] to the K record of BL_{en}.
(13) end for
(14) end for

Algorithm 4: Encryption algorithm (En).

Input: ID′, R'
Output: R
(1) i = ID′;
(2) $R = p'_{i1}, p'_{i2}, \ldots, p'_{im} + R'$;
(3) Return R

Algorithm 5: De algorithm.

After receiving the private key, the user i sends the parameter OP, the attribute set, and the signature to the AMS.

Step 2: The AMS verifies the signature sent by the user; if it is right, it will expose the OP.

Step 3: in the DO, define an access tree structure T'_{di} for the data di ready to be released;

Perform the attribute-based encryption algorithm encrypt function, input parameters (SMK, Di, Mdi′), encrypt plaintext data di′, and get ciphertext di″.

For access tree T_i, the node is x; the root node is R; the leaf node is z; the set of the leaf node is Z; and the pa (x) returns the parent node of the node x; num (x) returns the number of the node x, and the att (z) is returned to the corresponding attribute of the leaf node. Set the threshold value of node x as Kk_x ($0 < k_x < \text{num}_x$), and select the ($k_x - 1$) polynomial q_x for it. q_x (0) is the leaf node, which represents the secret of the node.

Random selection of secrets $s \in Z_p$; let $q_R(0) = s$, $q_x(0) = q_{\text{pa}(x)}(\text{num}(x))$; Encrypted ciphertext $CT = \{T, C_1 = M \cdot e(g, g)^{\alpha s}, C = g^{\beta s}, \forall Z \in Z_p, C_z^{(1)} = g^{q_{z(0)}}, C_z^{(2)} = g_{\text{att}(z)}^{xiq_z(1)}$. Send di″and signature to the AMS.

Step 4: the AMS will verify the signature after receiving di′, and the signature, if it passes, stores data di″.

4.2.3. Published in SNSP. The data file is published in the SNSP social network in the form of di″.

4.3. User Data Access

4.3.1. Visitor i Requests an Access Request. Visit i requests access to the AMS by sending requests and ID.

4.3.2. AMS Applies DO's Friend Relationship. After receiving the access request from user i, the AMS applies DO's friend relationship. (Let the parameter of the application friend relationship be t, where the parameter t is the parameter to determine the friend relationship between the AMS and DO. $T = 1$ means it is a close friend; otherwise, it is not a close friend.)

4.3.3. DO Retrieval Validation List V. If the ID of the user is retrieved in the list, it shows that the access user has already requested access to the data and is a close friend, so he jumps directly to Section 4.3.5 and sets it as a close friend.

If no access to the user ID is retrieved, it means that the user requests access to the data for the first time, so the process of encrypting the user ID is entered.

Step of encrypting user ID:

Step 1: enter the access user ID_i, BL_{in}.

Step 2: retrieve BL_{en} with user ID_i.

(i) If you get the query result, record R; this shows that the friend information is saved in the friend list, querying map[t], and determining that the ID of R in BL_{en} is the encrypted ID. Enter the next step.
(ii) If not found in the encrypted list (retrieval failure), it is not in the friend list, so it must not be a friend; the query result is returned, and this triggers the update of the user information when the next friend list is updated. Jump directly to step (5).
Step 3: output encrypted ID.

4.3.4. Query the Encrypted Data Table

Step 1: the DO server retrieves encrypted friend list.

Input: encrypted visitor's ID.

Retrieval. In the encrypted data list, find the input data and get the data after the data are returned, that is, the query result R.

Output: decryption the query result R' in the list.

Step 2: the DO server decrypts the query results.

In this paper, the De algorithm is used to decrypt the query results. The main process of the De algorithm is to generate pseudorandom sequence matrix FRS[$n * m$] using RanSee[0], which is received by the initialization stage. According to ID$'$, to determine the row in the pseudorandom sequence matrix, the pseudorandom code of the matrix is used to perform the XOR operation with R', and the final query result R is obtained, and the query result is returned to the DO.

The De algorithm process is shown in Algorithm 5.

4.3.5. Determine the Parameter Value. When the DO receives the R, it detects that if it is a close friend, it will be added to the verification list V.

According to the result of the query, set the value of parameter t, and send it to the AMS.

4.3.6. AMS Processing. The AMS gets the friend relationship value after receiving the parameter t and distributes different data according to the friend relationship.

If it is a close friend, then send the encrypted file di" and the AMS's parameters of the process of noise adding; If not, then send encrypted file di$'$.

In this way, if it is a common friend, it can obtain the encrypted data file after the first encryption through the decryption; if not, then it fails.

4.3.7. Visitors for Decryption Operations

Step 1: visitor received the result of the query.

Step 2: runs the CoA function to update the private key.

Step 3: if the user is a nonfriend, execute the (1); if the user is a common friend, execute (2), and the first encrypted data file is obtained; otherwise, if the user is a close friend, execute (1)(2), and the user can get the accurate data from the user.

(1) Using the private key decryptioned by the attribute set to get the file di$'$; execute function Decrypt (CT, SK) to decrypt: for each leaf node z, define a recursive function DecryptNode (CT, SK, z). If att$(z) \in S$, calculate DecryptNode (CT, SK, z) = $(e(\mathrm{Datt}_{(z)}{}^{(1)}, C_z{}^{(1)}))/(e(\mathrm{Datt}_{(z)}{}^{(2)}, C_z{}^{(2)})) = e(g,g)^{yqz(0)}$; if att $(z) \notin S$, then DecryptNode (CT, SK, z) = $\perp$. For every nonleaf node x, let its child node be ck, and k_x is used as a Lagrange interpolation node; calculate $e(g,g)^{yq_{c_k}(0)}$ $M = (Me(g,g)as) / (e(C,D) / e(g,g)^{yqz(0)}) = C_1 / (e(C,D)/e(g,g)^{ys})$.

(2) The user's exact data di are obtained by using encryption parameters.

4.4. Friend List Update

Step 1: add new friendship information.

New friendships include intimate relationships and nonintimate relationships, including new users who need to be recorded in the list in the process of accessing the user's query as well as the user's normal supplement to the information of their friends.

Step 2: delete the users who are not friends.

When the user relationship deteriorates, it will cancel the close friend relationship, so we will update the friend relationship.

Step 3: set the corresponding parameters.

If the intimate relationship has had a user deleted from a friend list, the verification list should also be modified.

4.5. Attribute Revocation. Attribute revocation mainly includes three parts, namely, generate re-encryption key, file re-encryption, and privilege replacement. When the DO revokes a number of user permissions to the file, the re-encryption key was first generated, and the file was re-encrypted with the key. After the user receives the feedback, the key is updated first. If the user is revoked by the DO, the file cannot be accessed. Otherwise, the user's access rights are not revoked.

Step 1: DO updates the user list URL that is needed to revoke permissions and add user value to the list; executing the function Grek to generate the re-encrypted private key; send URL table, re-encryption key, signature, and di$'$ to the AMS.

Step 2: AMS authentication signature.

If the signature is correct, the function Fre is executed to generate the re-encrypted ciphertext; the updated private key is sent to the user who is revoked; and then the URL list is updated after the completion of the transmission.

5. Proof of Security

5.1. Analysis of Privacy Security. The security of the social network privacy protection scheme is mainly to ensure that the AMS will not disclose the information of the encryption process and prevent the AMS from collusion with other illegal users. The security of the algorithm in PPSSN is demonstrated in detail in reference to [22].

5.1.1. The Security of the Algorithm

Theorem 1. *For algorithms En and De, according to the scheme to query operation $q_1, q_2, \ldots, q_n$, the scheme has query security when and only when $q_1, q_2, \ldots, q_m$'s joint information entropy is maximum.*

Proof. This paper is proved by the method of number induction.

When $m = 1$, a query operation is executed, and the record order of the friends list is randomly disturbed, and

the encrypted records change the sequence of the original list data and no longer correspond.

$p(q_1 = 1) = \cdots = p(q_1 = n) = 1/n$, $H(q_1) = H_{\max}(q_1) = \log n$. Therefore, the entropy of every query is the largest, so

$$H(q_1) = H(q_2) = H(q_m) = \log n. \tag{2}$$

When $m = 2$, since the read encrypted records are not related to the two query, Q1 and Q2 are independent for reading record operations, so there are $p(q_1, q_2) = p(q_1)p(q_2)$ and $H(q_1) = H(q_2) = \log n$. Then, each query is independent of each other.

Suppose $p(q_1, \ldots, q_m) = p(q_1)p(q_2)\ldots p(q_m)$; $H(q_1) = \ldots H(q_m) = \log n$, for the $m + 1$ query, the entropy of each query is the largest, and the queries are independent from each other; and $p(q_1, \ldots, q_m, q_m + 1) = p(q_1)p(q_2)\ldots p(q_m)p(q_{m+1})$:

$$H(q_1) = \cdots H(q_m) = H(q_{m+1}) = \log n. \tag{3}$$

We prove the following theorem. □

Theorem 2. *For algorithm En and De, the scheme satisfies nonrelevancy and nontraceability.*

Proof. Because when $\mathrm{BL}_{\mathrm{ori}}$ and $\mathrm{BL}_{\mathrm{en}}$ are represented as

$$\begin{aligned} \mathrm{BL}_{\mathrm{ori}} &= \{\mathrm{Rl}[1], \mathrm{Rl}[2], \ldots, \mathrm{Rl}[n]\}, \\ \mathrm{BL}_{\mathrm{en}} &= \{\mathrm{Rl}'[1], \mathrm{Rl}'[2], \ldots, \mathrm{Rl}'[n]\}. \end{aligned} \tag{4}$$

$\mathrm{BL}_{\mathrm{ori}}$ and $\mathrm{BL}_{\mathrm{en}}$ are the set of vectors. At this time, $\mathrm{Rl}[k]$ and $\mathrm{Rl}'[j]$ act as random vectors of $\mathrm{BL}_{\mathrm{ori}}$ and $\mathrm{BL}_{\mathrm{en}}$, respectively; then, the probability that the enemy can correctly obtain the mapping relationship between $\mathrm{Rl}[k]$ and $\mathrm{Rl}'[j]$ is $1/n$, and the information entropy of any query is

$$\begin{aligned} H(q) &= \sum_{t-1}^{m} p(xi) \log \frac{1}{p(xi)} = \sum_{t-1}^{m} \frac{1}{p(xi)} \log m \\ &= \log m. \end{aligned} \tag{5}$$

It can be seen that the information entropy of any single query is the largest.

Because each query is independent of each other, we know that the joint information entropy of n secondary query is

$$H(q_1 L q_n) = \sum_{i=1}^{m} H(q_n) = n \log m. \tag{6}$$

It can be obtained that the joint information entropy of n queries is the maximum.

From Theorem 1, we can get the query security of this scheme and satisfy the requirement that it is not related and cannot be traced. □

5.2. Confidentiality

5.2.1. Confidentiality of User Attribute Information. In this PPSNS scheme, the DO generates the user's private key and is responsible for the user's attribute revocation. In addition, the friend list is stored and updated by the DO, and the user's attribute information is only responsible for the DO, so the attribute information is confidential.

5.2.2. Confidentiality of Ciphertext Data. Differential privacy-based data file encryption can set parameters according to certain safety requirements, so it meets certain safety requirements.

5.2.3. Confidentiality of Attribute-Based Encryption Algorithm. The CP-ABE algorithm is proved to be safe under the standard model. The improvements made by this paper are as follows: the approach to generating the key and the phase of encryption. The above has proved the security of the improved algorithm, so the algorithm based on the attribute encryption is also safe.

5.2.4. Confidentiality of Pseudorandom Algorithm. After classifying the users' friends, we store our friend relationship in the DO's friend list. In the user access phase, the DO server will query the friend's list according to the user's access request to judge a friend's relationship, and the query operation of the DO server is not directly queried in the list of friends; however, the friend list $\mathrm{BL}_{\mathrm{en}}$ after the pseudorandom encryption and the DO server cannot directly access the friend list data $\mathrm{BL}_{\mathrm{ori}}$ of the DO. The security of the pseudorandom encryption algorithm has been proved in [18]. Therefore, we can see that the confidentiality of the friend list encryption algorithm is guaranteed in this scheme.

5.2.5. The Security of the Property Revocation Mechanism. When the attribute is revoked, the key will be re-encrypted, and the DO will generate s' randomly, generate a new access tree structure, and generate a new secret of restoring s. If the user has been revoked and cannot update the new property key, the updated cipher cannot be accessed because the new key is re-encrypted with the newly generated random number s'. Although the user's nonupdated $e(g, g)^{\alpha s}$ cannot be used to compute the new $e(g, g)^{\alpha s'}$, the revocation party in this case has forward security.

Assuming that the key generated by the re-encryption is leaked to the AMS, it is still unable to get a new secret, so it does not affect the confidentiality of the re-encrypted ciphertext, and the revocation scheme has a backward security.

5.2.6. Confidentiality of Access Control Policy. In the access control scheme, the re-encrypted operation is only AMS's leaf node operation on the access structure tree. It does not handle the internal nodes, so it cannot obtain the relational function associated with the internal nodes; thus, the access control strategy realizes confidentiality.

5.3. The Resistance of Collusion Attacks

5.3.1. Conspiracy Attack between Illegal Users and AMS. Because the user's private key generation, distribution, and verification are performed by the DO and AMS together, the

AMS is only responsible for verification and does not store private keys. Therefore, even if an unauthorized user is unauthorized to decrypt with the AMS, the private key cannot be illegally obtained, so the attacker cannot conspire to obtain the private key of the user.

5.3.2. Conspiracy Attack between the AMS and DO Server. In this paper, the encryption work based on differential privacy and the core steps based on attribute encryption are performed within the DO; the friends list is also stored in the DO, although the encryption process is in the DO server, but the DO server does not directly view the permissions of the friends list; although it can be verified by a given map[t] when decrypted, the friend parameter t used to communicate between the AMS and DO is not consistent with the friend parameters that are passed between the DO and DO servers. That is, the AMS cannot identify the classification parameters of the friends of the DO server. Therefore, the AMS cannot skip the verification of the DO to directly obtain the parameters of the friend relationship judgment in the DO server. So, although the DO is partially encrypted and distributed to the AMS and DO servers, the core privacy data are kept by the DO personally, which can prevent the AMS and DO servers from obtaining DO release data without DO authentication.

Because of this, illegal users are unable to illegally obtain decryption information with the AMS and DO servers. In sum, this paper can resist conspiracy attacks between AMS and DO servers and illegal users with AMS and DO servers.

5.3.3. Conspiracy Attacks of Illegal Users and DO Servers. The DO server does not store decryption related private keys, so illegal users cannot get information about decrypted data. There is no conspiracy attack between illegal users and DO servers.

6. Performance Analysis

6.1. Complexity Analysis. Compared with the EASiER [19] and scheme (I-WT-LPP) [21], this paper analyzes the complexity from four aspects: the overhead of the private key generation, the storage overhead of the user private key, the overhead of the DO to the data encryption, and the overhead of the decryption of the data by the user.

In the private key generation phase, each user must be considered in the EASiER scheme, so the complexity of the private key is $O(na)$, (n is the average number of users of social networks and a is the average number of attributes corresponding to the private key of the user). In the I-WT-LPP scheme, the DO generates only the main private key, so its complexity is $O(1)$. The PPSSN scheme in this paper does not generate a private key for each user, the complexity of this scheme is related to the number of attributes associated with the user's private key, which is $O(a)$.

For the storage of private keys, each user in the EASiER scheme needs to store the private key that the DO sends, so the complexity is $O(ma)$, (m is the average number of DO related to the user). In the I-WT-LPP scheme, users only store the private key and the private key of the DO distribution, so its complexity is $O(m) + O(a)$. In this paper, we also need to store the private key and private key issued by PPSSN for each user. The complexity of the scheme is $O(m) + O(a)$ for each user in the DO scheme.

D represents the size of the data file, and b indicates the average number of attributes required for encryption. The I-WT-LPP scheme performs three attribute encryptions on the attribute encryption, so its complexity is $O(D) + O(3b)$. The PPSSN encryption is complex in the phase of data encryption, and the schemes of EASiER and PPSSN adopt a double decker encryption mechanism, and their encryption complexity is $O(D) + O(b)$.

In the decryption phase of the user, similar to data encryption, since the three schemes are based on a hybrid encryption mechanism, the complexity of the EASiER schemes is $O(D) + O(c)$ where c represents the average number of attributes required to decrypt. The I-WT-LPP scheme performs three attribute encryptions on the attribute encryption, so its complexity is $O(D) + O(3c)$. The PPSSN encryption complexity is $O(D) + O(c)$.

6.2. Experimental Analysis

6.2.1. Experimental Environment. The experimental environment of this paper is as follows: Intel (R) Core (TM) i3-2370M, main frequency 2.40 GHz, 4.00 GB memory, 200 GB available disk space, Windows7 operating system, and the algorithm implementation code is written in C language. Among them, the attribute-based encryption algorithm is written based on cpabe-0.11 library [23]. In this paper, the scheme PPSSN is compared with the scheme (I-WT-LPP) [21] and the scheme AR-ABE [24]. By comparing the DO's processing consumption time and the storage consumption of each scheme under the same condition, it is possible to analyze the performance.

6.2.2. Experimental Results and Analysis. This paper analyzes the performance of the PPSSN scheme from the following three aspects. First, it analyzes the time consumption changes of the DO of the system with the increase in the number of access queries. Second, it analyzes the time consumption of the DO when the proportion of the close friends in the total number of users is different. Finally, it analyzes the changes in storage space required when the number of visitors is increased.

(1) Set the Number of Users to 100. The proportion of close friends is 10%, and the proportion of ordinary friends is 70%. Three groups of experiments were designed each time, and the average value of experimental results was taken. In Figure 4, the number of leaf nodes from structural tree T was 5, and in Figure 5, the number of leaf nodes of structural tree T was 10. The x-axis represents the number of requests per user access. The y-axis represents the DO time consumption of the system, and the data volume is 5 MB and 10 MB, respectively. When the data volume is

constant, the time spent on the DO side of the three schemes varies with the number of visitor queries, as shown in Figures 4 and 5.

As shown in Figure 4, when the number of leaf nodes is 5 and the data file is 5 MB, the overhead of the DO in the I-WT-LPP scheme and the AR-ABE scheme increased with the increase in the number of access queries. While the system overhead of the PPSSN scheme has not changed much when the number of access queries is increased, it will not increase significantly with the increase in access queries for a certain period of time. When the data file is 10 MB, the system overhead of this scheme is slightly higher than that of the other two schemes when the number of access queries is 1, but with the increase in the number of access queries, the overhead of the I-WT-LPP scheme and the AR-ABE scheme will increase, and the overhead of this scheme has not been greatly fluctuated.

As shown in Figure 5, when the number of leaf nodes is 10, the overhead of the three schemes is basically the same as when the number of leaf nodes is 5. This scheme has an obvious overhead advantage when the number of query times is greater than 1. At the same time, combined with Figures 4 and 5, we can see that when the number of access queries is certain, the cost of the DO of the system will also increase when the number of leaf nodes increases.

In sum, we know that when this scheme is compared with the I-WT-LPP and AR-ABE schemes, the number of access queries is 1, and the overhead of this scheme is slightly larger than the other two schemes; however, when the number of access queries is more than 1, the system overhead is maintained in a certain range, and it will not have an obvious increasing trend, and the advantage is obvious. Therefore, it shows that this scheme can provide better quality query services through the classification of friends with different users with different rights to provide, and it has a certain advantage over the other two schemes.

(2) Set the Number of Requests per User as Two. The total number of users is 100, of which the proportion of unfamiliar users is 10%. In Figure 6, the number of leaf nodes from the structural tree T is 5, and Figure 7 takes the number of leaf nodes of structural tree T to 10. The x-axis indicates the percentage of close friends in the total number of users. The y-axis represents the DO time consumption of the system, and the data volume is 5 MB and 10 MB, respectively. When the proportion of close friends in the total number of users is different, the time consumption changes of the query at the DO of the system are shown in Figures 6 and 7.

As shown in Figures 6 and 7, when the number of leaf nodes and the size of data files are certain and when the proportion of close friends increases, the time consumption of the system DO of the I-WT-LPP and AR-ABE schemes is basically unchanged, but the time consumption of the PPSSN scheme is significantly reduced, indicating that the implementation efficiency of this scheme is related to the ratio of close friends. In addition, because this scheme can store access records and reduce the cost of authentication in future access requests after receiving close friends access, the higher the ratio of intimate friends, the more obvious the superiority of this scheme is. At the same time, when the number of leaf nodes is increased, the overhead of the system is increasing; when the data file increases, the overhead of the DO increases, which is in accordance with the experimental results of the previous article.

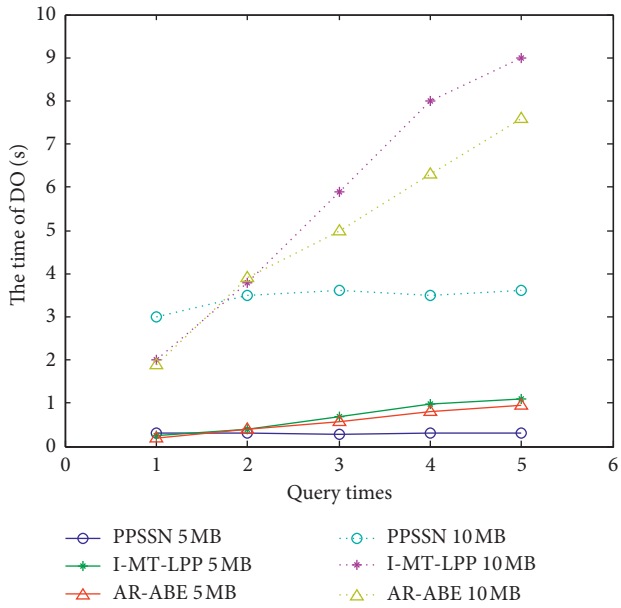

Figure 4: The time consumption of the DO when the leaf node is 5.

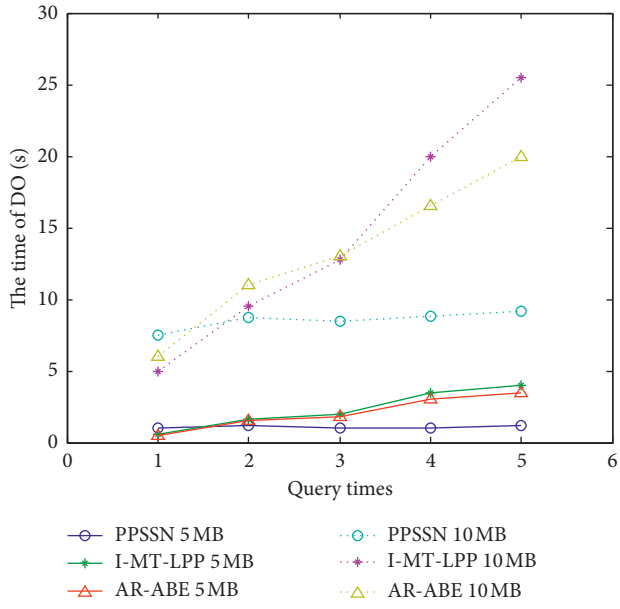

Figure 5: The time consumption of the DO when the leaf node is 10.

(3) Set the Number of Requests per User to Two. The number of leaf nodes is 5, the proportion of close friends is 10%, and the proportion of general friends is 70%. The x-axis represents the number of users accessing the social network, and the y-axis represents the actual storage space of the system. The amount of data is 20 MB and 50 MB,

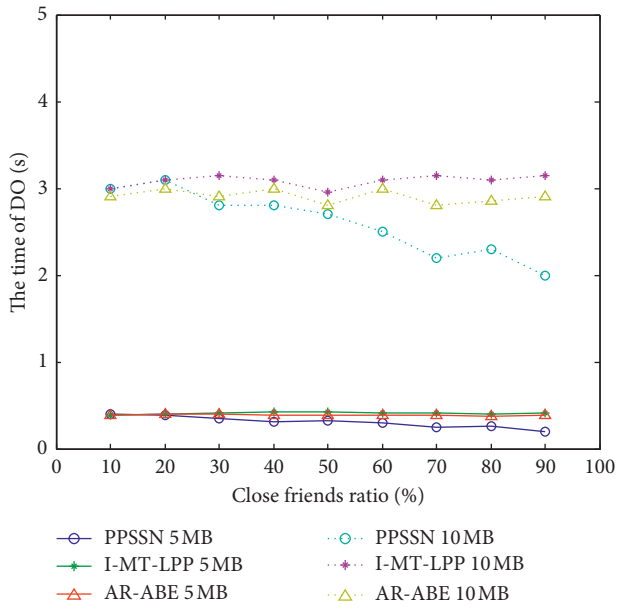

Figure 6: The time consumption of the DO when the leaf node is 5.

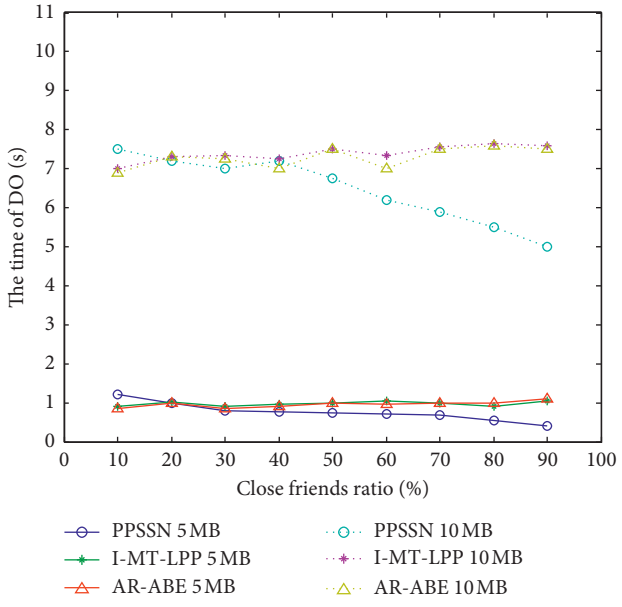

Figure 7: The time consumption of the DO when the leaf node is 10.

respectively. When the number of access users increases, the storage space of the system DO must be changed as shown in Figure 8.

As shown in Figure 8, when the number of data files of the leaf node number and the number of instances of user access is certain, this scheme is basically poor given the storage space occupied by the I-WT-LPP and AR-ABE schemes. Under the same conditions, the increase in the number of leaf nodes brings additional system storage cost, and the increase in data file size will also increase the storage cost of the system. When the number of leaf nodes and the size of the data file is certain, with the increase in the number of users, the three schemes increase the cost of the system at the same time, and the

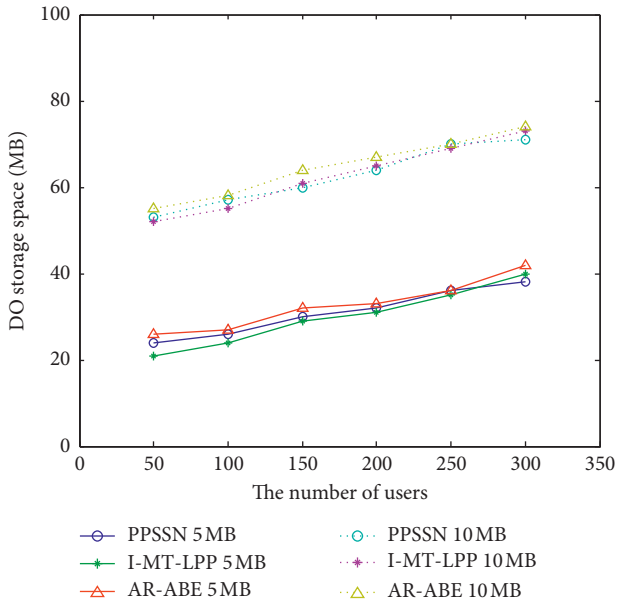

Figure 8: Changes in DO storage space.

increase in the amount of storage is not much different; this scheme adds a friend list to the DO and verifies the buddy relationship through a list, which adds a certain amount of overhead, but in the private key generation, the I-WT-LPP scheme and the AR-ABE scheme increase the storage overhead, so the total system overhead of the three cases is not much different. In this paper, the query efficiency is improved without additional storage overhead.

In sum, because the buffer mechanism is set up in the PPSSN scheme, the DO cost is greatly reduced when the number of visitors per capita increases. At the same time, the DO overhead of the PPSSN scheme is associated with the proportion of close friends, and as the proportion of close friends increases, the cost of the system is reduced as the access user is classified according to the intimacy. In addition, the storage cost of the system has not increased greatly, but it has been controlled within a certain range and is almost the same as the other two schemes. Therefore, PPSSN can improve the quality of service and provide fine-grained queries; meanwhile, the system overhead is basically unchanged, and there is a significant advantage.

7. Conclusion

In this paper, a social network privacy protection scheme is proposed. In the aspect of key generation, the identity of the user is verified by the AMS, and the property key is distributed to the legitimate user. The DO will generate the corresponding user key to its friends according to the friends list, and the visitors who get two keys can access the accurate information of the user according to the key. Since this scheme is classified according to the identity of the visitor, the data owner has a list of friends, and the user in the list can provide higher availability when the data are accessed on the platform. Under certain privacy protection levels, some users can improve the availability of data.

In addition, this scheme proposes a verification list V to reduce the subsequent access overhead to the users that have been visited. The buffer mechanism has an obvious advantage in the case of increasing query, and it is stored in the DO, so there is no risk of disclosing information.

Of course, this paper also has some shortcomings: while improving the quality of the query service and efficiency, the storage space would benefit from improvement, which will be considered in future research.

Data Availability

The all type data used to support the findings of this study have been deposited in the UCI Machine Learning Repository (http://archive.ics.uci.edu/ml/datasets.html). There are three data sets in this paper. The first data set is Amazon Commerce reviews set Data Set (http://archive.ics.uci.edu/ml/datasets/Amazon+Commerce+reviews+set). This data set mainly collects the online business evaluation of different users on Amazon website and records it as data set D1. The second data set is the credit card customer liquidated damages data set (http://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients). It makes statistics and collates the situation of customer liquidated damages in Taiwan and records it as data set D2. The third data set is an anonymous sample of access records to Amazon's internal resources (http://archive.ics.uci.edu/ml/datasets/Amazon+Access+Samples). The sampling time is from 0:00 on March 1, 2005, to 31:23 p.m. on August 31, 2010, and is recorded as data set D3.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant nos. 61572260, 61872196, 61872194, and 61402241, in part by the Jiangsu Natural Science Foundation for Excellent Young Scholar under Grant no. BK20160089, in part by Scientific & Technological Support Project of Jiangsu Province under Grant no. BE2017166, in part by Research of Natural Science of NJUPT under Grant no. NY217050, and in part by Jiangsu Government Scholarship for Overseas Studies.

References

[1] Z. Lv, H. Cheng, A. Chang, F. Dengguo, and C. K. Qu, "Privacy protection scheme for social network," *Journal of Communication*, vol. 35, no. 8, pp. 23–32, 2014.

[2] Y. Zhang, Y. Shen, H. Wang, Y. Zhang, and X. Jiang, "On secure wireless communications for service oriented computing," *IEEE Transactions on Services Computing*, vol. 11, no. 2, pp. 318–328, 2018.

[3] H. Wang, Z. Zhang, and T. Taleb, "Editorial: special issue on security and privacy of IoT," *World Wide Web*, vol. 21, no. 1, pp. 1–6, 2018.

[4] R. K. Lomotey, J. C. Pry, and C. Chai, "Traceability and visual analytics for the Internet-of-Things (IoT) architecture," *World Wide Web*, vol. 21, no. 1, 2018.

[5] F. Chen, Y. Luo, J. Zhang et al., "An infrastructure framework for privacy protection of community medical internet of things," *World Wide Web*, vol. 21, no. 1, 2018.

[6] Q. Huang, L. Wang, and Y. Yang, "DECENT: secure and fine-grained data access control with policy updating for constrained IoT devices," *World Wide Web-Internet & Web Information Systems*, vol. 11, pp. 1–17, 2017.

[7] A. Shamir, "Identity-based cryptosystems and signature schemes," in *LNCS 196: Proceedings of the Advances in Cryptology (CRYPTO)*, pp. 47–53, SpringerVerlag, Berlin, Germany, 1985.

[8] D. Boneh and M. Franklin, "Identity-based Encryption from the Weil pairing," in *Proceedings of the Advances in Cryptology-(CRYPTO 2001)*, pp. 213–2293, SpringerVerlag, Berlin, Germany, 2001.

[9] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *EUROCRYPT 2005, LNCS*, R. Cramer, Ed., vol. 3494, Springer, Heidelberg, Germany, pp. 457–473, 2005.

[10] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM Conference on Computer and Communications Security-CCS '06*, pp. 89–98, Alexandria, VA, USA, November 2006.

[11] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proceedings of the 2007 IEEE Symposium on Security and Privacy (SP '07)*, pp. 321–334, Washington, DC, USA, May 2007.

[12] B. Waters, "Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization," in *Proceedings of the 14th IACR International Conference on Practice and Theory of Public Key Cryptography (PKC 2011)*, pp. 53–70, Taormina, Italy, March 2011.

[13] M. Green, S. Hohenberger, and B. Waters, "Outsourcing the decryption of ABE ciphertexts," in *Proceedings of the USENIX Security Symposium*, vol. 3, Bellevue, WA, USA, August 2011.

[14] S. Yu, C. Wang, K. Ren, and W. Lou, "Attribute based data sharing with attribute revocation," in *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security-ASIACCS '10*, pp. 261–270, Sydney, Australia, March 2010.

[15] T. Naruse, M. Mohri, and Y. Shiraishi, "Attribute revocable attribute-based encryption with forward secrecy," *IPSJ Journal*, vol. 55, no. 10, pp. 2256–2264, 2014, in Japanese.

[16] Z. Liu, Z. F. Cao, and D. C. S. Wong, "Traceable CP-ABE: how to trace decryption devices found in the wild," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 1, pp. 55–68, 2015.

[17] Z. Liu, Z. F. Cao, and D. C. S. Wong, "Blackbox traceable CP-ABE: how to catch people leaking their keys by selling decryption devices on ebay," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security-CCS '13*, pp. 475–486, New York, NY, USA, November 2013.

[18] R. Baden, A. Bender, N. Spring, B. Bhattacharjee, and D. Starin, "Persona: an online social network with user-defined privacy," in *Proceedings of the ACM SIGCOMM 2009 Conference on Data Communication (SIGCOMM 2009)*, pp. 135–146, Barcelona, Spain, August 2009.

[19] S. Jahid, P. Mittal, and N. Borisov, "EASiER: encryption-based access control in social networks with efficient revocation," in *Proceedings of the 6th ACM Symposium on Information,*

Computer and Communications Security (ASIACCS 2011), pp. 411–415, HongKong, China, March 2011.

[20] X. Liang, X. Li, R. Lu, X. Lin, and X. Shen, "An efficient and secure user revocation scheme in mobile social networks," in *Proceedings of the International Conference on Global Telecommunications Conference (GLOBECOM 2011)*, Houston, TX, USA, December 2011.

[21] X. Lin, Y. Han, K. Yan, and X. Yang, "Location privacy preserving scheme against attack from friends in SNS," *Journal of Communication*, vol. 37, no. S1, pp. 224–230, 2016.

[22] S. T. Yang, C. G. Ma, and C. L. Zhou, "Privacy protection model and scheme for LBS," *Journal of Communication*, vol. 35, no. 8, pp. 116–124, 2014.

[23] J. Bethencourt, A. Sahai, and B. Waters, "Advanced crypto software collection: the cpabe toolkit [EB/OL]," March 2011, http://acsc.cs.utexas.edu/cpabe/.

[24] T. Naruse, M. Mohri, and Y. Shiraishi, "Attribute revocable attribute-based encryption with forward secrecy for fine-grained access control of shared data," *IEICE Technical Report Information & Communication System Security*, vol. 114, no. 10, pp. 181–186, 2015.