IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Volume-3, Issue-9 E-ISSN: 2347-2693

Defend Clandestine Rated Multi-keyword Search for Multi-Proprietors in Cloud Computing

Vanishree K.A.[1*], Santhiya. C[2] and M.K. Chandrasekaran PhD[3]

[1*,2,3] *Dept. Of Computer Science and Engineering, Anna University, India,*

www.ijcseonline.org

Received: Sep/02/2015 Revised: Sep/11/2015 Accepted: Sep /27/2015 Accepted: Sep/30/2015

Abstract—Due to the escalating popularity of cloud computing, more and more data owners are encouraged to outsource their data to cloud servers for great convenience and scaled-down the cost in data management. For privacy concerns, secure searches over encrypted cloud data have motivated numerous research works under the solitary owner model. However, most cloud servers in practice do not just serve single owner; instead, they support multiple owners to share their profits fetched by cloud computing. In our proposed scheme which is deals with Defend Clandestine rated Multi-keyword Search in a Multi-proprietor model (DCMSM). To activate cloud servers to perform secure search without knowing the actual data of both keywords and trapdoors, we methodically build a newfangled secure search protocol. We propose another technique as distinct Additive Order and Privacy Preserving Function family for the purpose of rank the search results and preserve the privacy of appositeness scores between keywords and files. By introducing a modish dynamic secret key generation protocol and a novel data user authentication protocol to prevent the attackers from intruding secret keys and concealing the legal data users submitting searches. Additionally, DCMSM supports efficient data user invocation. Comprehensive experiments on real-world datasets confirm the efficaciousness and efficiency of DCMSM.

Keywords—Cloud Computing, Rated Keyword Search, Multiple Owners, Privacy Preserving, Dynamic Secret Key.

I. INTRODUCTION

Cloud computing is a dissident technology that altering the way IT hardware and software are designed and purchased. Cloud computing provides abundant benefits including easy access, decreased costs, quick deployment and flexible resource management, etc. Enterprises of all sizes can leverage the cloud to boost up the innovation and collaboration. Even though cloud computing lagging on privacy concerns, individuals and enterprise users is hesitate to outsource their sensitive data, including emails, personal health records and government confidential files, to the cloud. This is because only sensitive data are outsourced to a remote cloud; the corresponding data owners lose direct control of these data.

Cloud service providers (CSPs) would assure to ensure owners' data security using various mechanisms like virtualization and firewalls. But, these mechanisms does not protect owners data privacy the CSP itself, since the CSP take over full control of cloud hardware, software, and owners' data. Encryption on sensitive data before outsourcing can preserve data privacy against CSP. However, a very challenging problem is data encryption makes the traditional data utilization service based on plaintext keyword search. A petty solution to this problem is to download all the encrypted data and decrypt them locally. However, this method is obviously not practical because it will cause a massive amount of communication overhead. Most cloud servers in practice does not just serve single owner; instead, they support multiple owners to share their profits fetched by cloud computing. For example, in health care service some volunteer patients would agree to share their health data on the cloud. To assist the government in making a satisfactory policies on health care service, or to help medical institutions conduct useful research, some volunteer patients would agree to share their health data on the cloud. To preserve their privacy, they will encrypt their own health data with their secret keys. In this scenario, only the authorized organizations can perform a secure search over this encrypted data contributed by multiple data owners. Such a Personal Health Record sharing system, where multiple data owners are involved, can be found at mymedwall.com.

Compared with the single-owner scheme, developing a full-fledged multi-owner scheme will have many new challenging problems. First, in the single-owner scheme, the data owner wants to stay online to generate trapdoors for data users. However, when a huge amount of data owners are presented, asking them to stay online immediately to generate trapdoors would seriously affect the flexibility and usability of the search system. Second, since nobody would be willing to share our secret keys with others, different data owners would prefer

to use their own secret keys to encrypt their secret data. At the same time, it is very challenging to perform a secure, convenient, and efficient search over the data encrypted with different secret keys. Third, when multiple data owners are involved, we should ensure efficient user enrollment and revocation mechanisms, so that our system delights in excellent security and scalability.

In our proposed scheme which is deals with Defend Clandestine rated Multi-keyword Search in a Multi-proprietor model (DCMSM). The main contributions of this paper are listed as follows:

- We define a multi-proprietor model for privacy preserving keyword search over encrypted cloud data.
- We propose an efficient data user authentication protocol, which not only prevent the attackers from intruding secret keys and concealing the legal data users submitting searches, but also enables data user authentication and revocation.
- We systematically construct a newfangled secure search protocol, which not only activate cloud servers to perform secure search without knowing the actual data of both keywords and trapdoors, but also allows data owners to encrypt keywords with self-chosen keys and allows authenticated data users to query without knowing these keys.
- We propose an Additive Order and Privacy Preserving Function family (AOPPF), according to their preference which allows data owners to protect the privacy of appositeness scores using different functions while still permitting the cloud server to rank the data files accurately.
- We conduct Comprehensive experiments on real-world datasets confirm the efficaciousness and efficiency of DCMSM.

II. PROBLEM FORMULATION

We first define,

i) System model

ii) Threat model.

2.1 System Model

In our multi-proprietor and multi-user cloud computing model, four units are involved, as illustrated in Fig. 1; they are

1) Data owners,

2) The cloud server,

3) Administration server, and

4) Data users.

In fig. 1 Data owners have a collection of files F. Data owners first construct a secure searchable index I on the keyword set W extracted from F, and then they submit I to the administration server. At last, data owners encrypt their files F and outsource the corresponding encrypted files C to the cloud server. After receiving I, the administration server decrypts I for the authenticated data owners and outsources the decrypted index to the cloud server. Initially, the data user computes the corresponding trapdoors and submits them to the administration server if they want to search t keywords over these encrypted files stored on the cloud server.

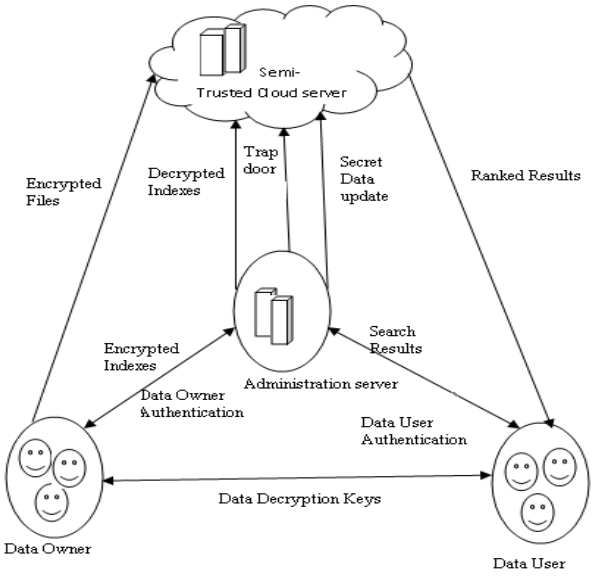

Fig. 1: Architecture of Defend Clandestine rated Multi-keyword Search in a Multi-proprietor model.

The administration server will further decrypt the trapdoors and submit them to the cloud server, which is happen once the data user is authenticated by the administration server. The cloud server searches the encrypted index I of each data owner and returns the corresponding set of encrypted files after receiving the trapdoor T.To enhance the file retrieval accuracy and save communication price, a data user would tell the cloud server a parameter k and cloud server would return the top-k relevant files to the data user. Once they receive the top-k encrypted files from the cloud server, they can able to decrypt these returned files.

2.2 Threat Model

In our threat model, we assume the administration server is trustworthy. The administration server can be any trusted third party, e.g., the Certificate Authority in the Public Key Infrastructure, the aggregation and

distribution layer and the third party auditor. Data owners and data users who passed the authentication of the administration server are also trusted. Conversely, the cloud server is not trusted. Instead, we treat the cloud server as 'curious but honest' .The cloud server reviews our proposed protocol, but it is keen to gain the contents of encrypted files, keywords, and appositeness scores.

2.3 Design Goals and Security Definitions

To enable Defend Clandestine rated Multi-keyword Search in a Multi-proprietor model in cloud environment, our system design should concurrently satisfy security and performance goalmouths.

• **Ranked Multi-keyword Search over Multi-proprietor:** It should permit multi-keyword search over encrypted files which would be encrypted with different keys for different data owners. To allow the cloud server to rank the search results among different data owners and return the top-k results is also needed.

• **Data owner scalability:** The proposed scheme should allow new data owners to enter this system without heartrending other data owners or data users, i.e., the scheme should support data owner scalability in a plug-and-play model.

• **Data user cancellation:** The proposed scheme that only authenticated data users can perform correct searches is to be ensured. There is no longer performing correct searches over the encrypted cloud data if the data user is revoked at once.

• **Safekeeping Goals:** The security goals achieve the following security goals:

1) Keyword Semantic Security will prove that DCMSM achieved semantic security contrary to the chosen keyword attack.

2) Keyword secrecy since the adversary A can know whether an encrypted keyword matching the trapdoor, we customize the enervated security goal i.e., we should ensure that the probability for the adversary A to infer the actual value of a keyword is negligibly more than randomly guessing.

3) Appositeness score secrecy should ensure that the cloud server cannot infer the actual value of the encoded appositeness scores.

III. PRELIMINARIES

We introduce some techniques and our detailed construction.

3.1 Bilinear Map

Let G and G1 denote two cyclic groups with a prime order p. We further denote g and g1 as the generator of G and G1, respectively. Let ê be a bilinear map ê: G×G → G1, then the following three conditions are satisfied:

S.no	Techniques	Eqautions
1.	Bilinear	$\forall a, b \in Zp$, $\hat{e}(ga , gb) = \hat{e}(g, g)ab$.
2.	Non-degenerate	$\hat{e}(g, g) \neq 1$
3.	Computable	Ê can be efficiently computed.

Table. 1: Bilinear Map.

3.2 Bilinear Diffie-Hellman Problem and Bilinear Diffie-Hellman Assumption

The Bilinear Diffie-Hellman (BDH) problem in (G, G1 , ê) is described as follows,

Given :	Random $g \in G$, and ga , gb , gc for some $a, b, c \in Zp$.
To compute:	$\hat{e}(g, g)abc \in G1$.

Table. 2: Computation of BDH problem.

The BDH assumption is presented as follows,

Given:	(G, G1 , ê), $g \in G$, and ga , gb , gc for some $a, b, c \in Zp$.
To compute:	$Pr[A(g^a , g^b , g^c) = \hat{e}(g, g)^{abc}] \geq \epsilon$.

Table. 3: BDH Assumption.

The BDH assumption tells that the advantage ϵ is negligible for any polynomial time A.

IV. DATA USER AUTHENTICATION

The administration server decrypts trapdoors for data users to prevent attackers from pretending to be legal data users performing searches and launching statistical attacks based on the search result, data users must be authenticated. Traditional authentication methods often follow three steps.

1) Data requester and data authenticator share a secret key, say, k_0.

2) The requester encrypts his personally identifiable information d_0 using k_0 and sends the encrypted data (d_0) k to the authenticator.

3) The authenticator decrypts the received data with k_0 and authenticates the decrypted data.

Even though this method has two main shortcomings:

1) To unchanged the remaining secret key shared between the requester and the authenticator so that it is easy to incur replay attack.

2) One time the secret key is revealed to attackers, the authenticator cannot differentiate between the legal requester and the attackers; the attackers can pretend to be legal requesters without being detected. In this, we first give an overview of the

data user authentication protocol. Then, we commence how to achieve secure and efficient data user authentication. Finally, we demonstrate how to detect illegal searches and how to enable secure and efficient data user revocation.

4.1 Overview

The main idea of the user authentication protocol is illustrated by an example. Assume Alice wants to be authenticated by the administration server, so she starts a conversation with the bob (server). The contents of the conversation are authenticated by bob. Once the contents are authenticated, both Alice and the bob will generate the initial secret key according to the conversation contents. After the initialization, to be authenticated successfully, Alice has to provide the historical data of their conversations. If the authentication is successful, both Alice and the administration server will change their secret keys according the contents of the conversation. In this way, the secret keys keep changing dynamically; without knowing the correct historical data, an attacker cannot start a successful conversation with the administration server.

4.2 User Authentication

We first introduce the format of the authentication data dynamic key generation method and the authentication protocol. As shown

Request counter	Last Request Time	Personally Identifiable Data	Random Number	CRC

Table. 4: Format of Authentication Data

In table. 4, the authentication data consists of five parts:

1) The request counter field records the number of search requests that the data user has submitted.

2) The last request time field asks the data user to provide the historical data of his previous request time.

3) The personally identifiable data (e.g., passport number, telephone number) field is used to identify a specific data user, while the random number and CRC field are added.

4) The random number and CRC field further used to check whether the authentication data has been tampered.

The key point of a successful authentication is to offers

- The dynamically changing secret keys.
- The historical data of the corresponding data user.

Let $k_{i,j}$ indicates the secret key shared between administration server and the jth data user Uj after i instances of search requests, and $d_{i,j}$ denotes the authentication data for the (i + 1)th request of Uj . Our authentication protocol runs in the following six steps.

A. Data user Uj prepares his authentication data $d_{i,j}$, i.e., Uj needs to fill in all the fields of authentication data based on his historical data.

B. Data user Uj encrypts $d_{i,j}$ with the current secret key $k_{i,j}$ and submits the encrypted authentication data $(d_{i,j})_{k_{i,j}}$ to the administration server.

C. The data user Uj generates another secret key $k_{i+1,j} = k_{i,j} \oplus H(d_{i,j})$, and stores both $k_{i,j}$ and $k_{i+1,j}$ after the authenticated data is submitted.

D. After receiving Uj's encrypted authentication data, the administration server decrypts it with $k_{i,j}$.

E. The administration server verifies the request counter, last request time, personally identifiable data and CRC, respectively. If the authentication succeeds, the administration server first generates a new secret key $k_{i+1,j} = k_{i,j} \oplus H(d_{i,j})$, then he replies a confirmation data $d_{i+1,j}$, and encrypts it with $k_{i+1,j}$. Otherwise, the administration server encrypts $d_{i+1,j}$ with secret key $k_{i,j}$.

F. After receiving a reply from the administration server, the data user Uj will try to decrypt it with $k_{i+1,j}$. If the decrypted data contains the affirmation data, the authentication is successful. Or else, the authentication is observed as being unsuccessful. The data user deletes the new generated secret key $k_{i+1,j}$ and considers whether to start another authentication or not.

After each successful authentication process, the secret key will be changed dynamically according to the previous key and some historical data through the successful authentication between the administration server and the data user. Therefore, once an attacker steals a secret key, he can hardly get any benefits and hack the user data in easy manner. On one hand, the attackers cannot even construct a legal authentication data if the attacker knows nothing about the historical data of the legal data user. On the other hand, if the previous key will be expire, and then the legal data user performs another successful authentication.

4.3 Illegal Search Detection

In our proposed scheme, the authentication process is sheltered by the dynamic secret key and the historical information. If an attacker has successfully eavesdropped the secret key and then he starts to construct the authentication data. If the attacker has not successfully eavesdropped the historical data, e.g., the request counter, the last request

time, he cannot construct the correct authentication data.

Therefore this illegal action will soon be detected by the administration server. Further, if the attacker has successfully eavesdropped all data and the attacker can correctly construct the authentication data and pretend himself to be without being detected by the administration server. However, once the legal data user performs his search, since the secret key on the administration server side has changed, there will be contradictory secret keys between the administration server and the legal data user. Therefore, the data user and administration server will soon detect this illegal action.

4.4 Data User Revocation

Data user revocation does not require decrypting and updating large amounts of data stored on the cloud server. Inspite of, the administration server only needs to update the secret data S_a stored on the cloud server.

$$S_a = g^{ka1 \cdot ka2 \cdot ra}$$

Where,

ka1 and ka2 are the secret keys of the administration server, and

ra is randomly generated for every update operation.

As a result, the previous trapdoors will be expired without the help of the administration server; the revoked data user cannot generate the correct trapdoor Tw h′. Therefore, once the data user is revokes, then he cannot perform correct searches.

V. MATCHING DIFFERENT-KEY ENCRYPTED KEYWORDS

Many data owners are often involved in practical cloud applications. They would be unwilling to share secret keys with others because of privacy concerns. Instead, they prefer to use their own secret keys to encrypt their sensitive data. When keywords of different data owners are encrypted with different secret keys, the upcoming queries from the data user are how to locate different-key encrypted keywords among multiple data owners. To enable secure, efficient and convenient searches over encrypted cloud data owned by multiple data owners. We systematically design schemes to achieve the following three requirements:

1) Different data owners use different secret keys to encrypt their keywords.

2) Without knowing these secret keys the authenticated data users can generate their trapdoors.

3) Upon receiving trapdoors, without knowing the actual value of keywords or trapdoors the cloud server can find the corresponding keywords from different data owners encrypted keywords.

5.1 Overview

Assume Alice wants to use the cloud to store her file; she first encrypts her file, and gets the ciphertext. To enable other users to perform secure searches on ciphertext, Alice extracts a keyword, and sends the encrypted keyword to the administration server. The administration server further decrypts the keyword, and submits the keyword to the cloud server. Now Bob wants to search a keyword, he first generates the trapdoor and submits it to the administration server. The administration server decrypts the trapdoor, generates a secret data, and submits trapdoor, secret data to the cloud server. The cloud server will judge whether Bob's search request matches Alice's encrypted keyword by checking whether those are holds or not.

5.2 Construction Initialization

Our construction is based on the abovementioned bilinear map. Let g and g1 denote the generator of two cyclic groups G and G1 with order p. Let ˆe be a bilinear map ˆe : G × G → G1. Given different secret parameters as input, a randomized key generation algorithm will output the private keys used in the system.

ka1 ∈ Z+p , ka2 ∈ Z+p , ki;f ∈ Z+p , ki;w ∈ Z+p ←(0, 1) ∗ where ka1 and ka2 are the private keys of the administration server. ki;w and ki;f are the private keys used to encrypt keywords and files of data owner Oi. Let H(·) be a public hash function, its output locates in Z+p.

5.3 Keyword Encryption

For keyword encryption, the following conditions should be satisfied:

1) For encrypting the keywords different data owners use their own secret keys.

2) It would be encrypted to different cipher-texts each time for the same keyword.

These properties benefit our scheme for two reasons.

1) Losing the key of one data owner would not lead to the acknowledgement of other owner's data.

2) The cloud server cannot see any relationship among encrypted keywords.

The data owner delivers keywords to the administration server, and the administration server further decrypts the keyword with his secret keys and gets the result. Therefore the administrative server further submits encrypted keyword to the cloud server. Since the administration server only does simple computations on the encrypted data, he cannot learn any sensitive information from these random encrypted data without knowing the secret keys of data owners.

5.4 Trapdoor Generation

Our proposed scheme should satisfy two main conditions to make the data users generate trapdoors securely, conveniently and efficiently. They are,

1) The data user does not need to ask a large amount of data owners for secret keys to generate trapdoors.

2) For the same keyword, the trapdoor generated each time should be different. To meet this condition, the trapdoor generation is conducted in two steps:

2.1) Based on searching keyword and a random number the data user generates trapdoors.

2.2) The administration server decrypts the trapdoors for the authenticated data user.

Assume a data user wants to search keyword, so he encrypts it as follows. Secret keys of data owners are not required during the trapdoor generating process. Moreover, with the help of random variable for the same keyword we can generate two different trapdoors which prevent attackers from knowing the relationship among trapdoors. Upon receiving trapdoors, the administration server first generates a random number and then decrypts the trapdoor. Finally, the administration server submits trapdoor to the cloud server.

5.5 Keywords Matching among Different Data Owners

All encrypted files and keywords of different data owners are stored in the cloud server. The administration server will also store a secret data on the cloud server. The cloud will search over the data of all these data owners after receiving a query request. The cloud processes the search request in two steps. First, the cloud matches the queried keywords from all keywords stored on it, and it gets a candidate file set. Second, the cloud ranks files in the candidate file set and finds the most top-k relevant files. We introduce the matching strategy here, while leaving the task of introducing the ranking strategy. When the cloud obtains the trapdoor and encrypted keywords. Then he can judge whether an encrypted keyword is located and holds if the process is true.

VI. PRIVACY PRESERVING RANKED SEARCH

However, which is impossible to simply return undifferential files to data users for the following two reasons. First, returning all candidate files would cause plentiful communication overhead for the whole system. Second, according to their query the data users would only concern the top-k relevant files. In this, we first clarify an order and privacy preserving encoding scheme. Then we exemplify an additive order preserving and privacy preserving encoding scheme. Finally, we apply the proposed scheme to encode the appositeness scores and obtain the top-k search results.

X	1	2	3	4	5
f(x)	100-1000	1100-1800	2000-4200	4300-5000	5100-7000

Table. 5: An example of Order Preserving and Privacy Preserving Function

6.1 Order and Privacy Preserving Function

To rank the appositeness score while preserving its privacy, the proposed function should satisfy the following conditions. 1) According to the encoded appositeness scores, the function should preserve the order of data, through this the cloud server determine which file is more relevant to a certain keyword.2) This function should not be cancelled by the cloud server so that cloud server can make comparisons on encoded appositeness scores without knowing their actual values. 3) Dissimilar data owners should have different functions such that revealing the encoded value of a data owner would not lead to the leakage of encoded values of other data owners. In order to satisfy condition 1, we introduce a data processing part which preserves the order of data files. To satisfy condition 2, we introduce a disturbing part which helps prevent the cloud server from revealing this function. To satisfy condition 3, we use the process ID of data owners.

6.2 Additive Order and Privacy Preserving Function

Obviously, from table f(x) is order preserving; to preserve privacy, a disturbing part is introduced. To evaluate the secure ranked multi-keyword search, the sum of any two encoded appositeness scores should still be ordered and privacy preserved.

6.3 Encoding appositeness scores

It encodes the appositeness score with an additive order and privacy preserving function in our additive order and privacy preserving function family. Through this data owners can individually choose a function from this family to protect the privacy of their appositeness scores.

6.4 Ranking search results

We implement the sum of the appositeness scores between the no. of files and matched keywords for data owner collection as the metric to rank search results. The cloud server ranks the sum of encoded appositeness score with the following two conditions:

(1) Two encoded data belong to the same data owner.
(2) Two encoded data belong to two different data owners. Thus, it is easy for the cloud to return the top-k relevant files to the data user.

VII. SECURITY ANALYSES

It define as follows,

7.1 Data Files

Before uploading the data files are protected by symmetric encryption. Since the encryption algorithm is not breakable

and also the cloud server cannot know the data.

7.2 Keywords

We originate the security goals achieved by DCMSM with the semantically secure against the chosen keyword attack under the selective security model and achieve keyword secrecy in the random oracle model.

7.3 Trapdoors

The cloud server must solve the discrete logarithm problem with large prime factor if they want to know the actual value of the trapdoor, and differentiate two trapdoors. Through this the privacy of trapdoor is protected as long as the discrete logarithm problem is rigid.

7.4 Appositeness Scores

Appositeness scores are encoded with two Additive Order and Privacy Preserving Functions and we analyze the security of additive order and privacy preserving functions. By using this technique and data owner ID to get the value and form the N values. Through this it is infeasible to break the additive order and privacy preserving family function and encoded relevance score also preserved.

VIII. PERFORMANCE EVALUATION

8.1 Evaluation Settings

We use Internet Request For Comments dataset (RFC) for conducting performance experiments on a real data set. To extracting keywords from each RFC file we use Hermetic Word Frequency Counter. We examine the keyword statistics such as the keyword frequency in each file, the length of each file, the no. of files containing a specific keyword, etc after the keyword extraction also calculate the appositeness score of a keyword to a file based on these statistics. The file size and keyword frequency of this data set can also be estimated.

The experiment programs are coded using the ASP.net on a PC with 2.2GHZ Intel Core CPU and 2GB memory. We implement all necessary routines for data owners to preprocess data files: for the data user to generate trapdoors, for the administrative server to decrypt keywords, trapdoors, and for the cloud server to perform ranked searches.

8.2 Evaluation Results

Evaluation Techniques	Process	Time utilization	No. of keywords usage
Index Construction	It use different datasets with same dictionary=4000, & for same dataset with different dictionary size=1000.	0.1s -1s	1000-10000
Trapdoor Generation	Use different size of keyword with the same no of queried keywords=100.	0.026s-0.031s	100-1000
Decryption by the administration server	Evaluate the time cost of same average no of keywords per owner.	3.34s	Each data owner has 100 keywords.
Search	Pairing the keywords.	According to the file size.	100-2000

Table. 6: Time cost of index construction, generating trapdoors, administration server and search.

IX. RELATED WORKS

9.1 Searchable Encryption

The proposed scheme to encrypt each word in a file independently and allow the server to find whether a single queried keyword is contained in the file without knowing the exact word.

9.2 Secure Keyword Search in Cloud Computing

The previous work only consider one data owner, through this the data owner and data users can easily communicate and exchange secret information. At the same time, the numerous data owners are involved in the system; secret information exchanging will cause significant communication overhead.

CONCLUSION

When comparing with previous works, our schemes enable authenticated data users to achieve secure, convenient, and efficient searches over multiple data owner's data. To activate cloud servers to perform secure search without knowing the actual data of both keywords and trapdoors, we methodically build a newfangled secure search protocol. We propose another technique as distinct Additive Order and Privacy Preserving Function family for the purpose of rank the search results and preserve the privacy of appositeness scores between keywords and files. By introducing a modish dynamic secret key generation protocol and a novel data user authentication protocol to prevent the attackers from intruding

secret keys and concealing the legal data users submitting searches. As our future work, we will consider the problem of secure fuzzy keyword search in a multi-owner paradigm.

REFERENCES

[1] C. Wang, S. S. Chow, Q. Wang, K. Ren, and W. Lou, Privacy-preserving public auditing for secure cloud storage," Computers, IEEE Transactions on, vol. 62, no. 2, pp. 362-375, **2013**.

[2] D.Song, D.Wagner, and A.Perrig, "Practical techniques for searches on encrypted data," in Proc. IEEE International Symposium on Security and Privacy (S&P'00), Nagoya, Japan, Jan.**2000**, pp. 44-55.

[3] R.Curtmola, J.Garay, S.Kamara, and R.Ostrovsky, "Searchable symmetric encryption: improved definitions and efficient constructions," in Proc. ACM CS'06, VA, USA, Oct. **2006**, pp.79-88.

[4] N.Cao,C. Wang, M. Li, K. Ren, and W. Lou, "Privacy-preserving multi-keyword ranked search over encrypted clouddata,"inProc.IEEEINFOCOM'11,Shanghai,China,Apr.**2011**,pp. 829-837.

[5] N.Cao, C.Wang, M. Li, K. Ren, and W. Lou,"Privacy-preserving multi-keyword ranked search over encrypted cloud data," Parallel and Distributed Systems, IEEE Transactions on,vol. 25, no. 1, pp. 222-233, **2014**.

[6] T. Jung, X. Y. Li, Z. Wan, and M. Wan, "Privacy preserving cloud data access with multi-authorities," in Proc. IEEE INFOCOM'13, Turin, Italy, Apr. **2013**, pp. 2625-2633.

