An Efficient Queries Processing Model Based on Multi Broadcast Searchable Keywords Encryption (MBSKE)

Belal Ali Al-Maytami[1], Pingzhi Fan[1], Abir Jaafar Hussain[2], Thar Baker Shamsa[2] and Panos Liatsis[3]

[1] Institute of Mobile communication, Southwest Jiaotong University, Chengdu, China
[2] Liverpool John Moores University, Department of Computer Science, Liverpool, L33AF, UK
[3] Department of Computer Science, Khalifa University of Science and Technology, Abu Dhabi, UAE

Abstract

Cloud computing is a technology which has enabled many organizations to outsource their data in an encrypted form to improve processing times. The public Internet was not initially designed to handle massive quantities of data flowing through millions of networks. So the rapid increase of broadcast users and the growth of the amount broadcasted information leads to slow sending quires and receiving encrypted data from the cloud. In order to solve this problem Next Generation Internet (NGI) is developed with high speed, while keeping the privacy of data. This research proposes a novel search algorithm called Multi-broadcast Searchable Keywords Encryption, which processes queries having a set of keywords. This set of keywords is sent from the users to the cloud server in an encrypted form, thus hiding all information about the user or the content of the queries from the cloud server. The proposed method uses caching algorithm and provide an improvement of 40% in terms of runtime and trapdoor. In addition, the method minimizes computational costs, complexity, and maximizes throughput, in the cloud environment, whilst maintaining privacy and confidentiality of both the user and the cloud. The cloud returns encrypted query results to the user, where data is decrypted using the users' private keys.

Keywords: Data encryption, Cryptography, Coding and information theory

1. INTRODUCTION

Cloud services based on cloud servers gained major popularity in recent years. Benefits of cloud servers include scalable and elastic storage and computation resources through the internet to users. Among the main features of outsourcing data services, i.e., cloud infrastructures are physically hosted and maintained by the cloud servers to reduce the risk and to hide information [1]. Furthermore, cloud computing and Next Generation Internet (NGI) are still concerned around personal data and transparency. They provide better services and greater data sharing. Focus on designing a new shape of internet, re-imagine and re-engineer the procedures of transparency, privacy, cooperation, and protection of data.

With NGI the huge number of users who will move online will look very different from the current number of users. In the future, we need to take into consideration some solutions related to the traffics and security of data through transferring the huge data, design new models to solve and manage the increasing number of users.

In the general broadcast encryption system, the server broadcasts the encrypted information to the users in the system, and any user can listen to the broadcast to obtain the encrypted information. Only the user in the authorized user set can decrypt with the private key broadcast ciphertext, to restore the corresponding clear text information. If all unauthorized users collide with the broadcast information, the broadcast encryption system has complete anti-conspiracy characteristics. At present, broadcast encryption as a commonly used encryption means has been widely used in pay TV, digital rights management, satellite communications, video teleconferencing, wireless sensor networks [2], health management system (HMS) [3], and geographical information system (GIS) [4].

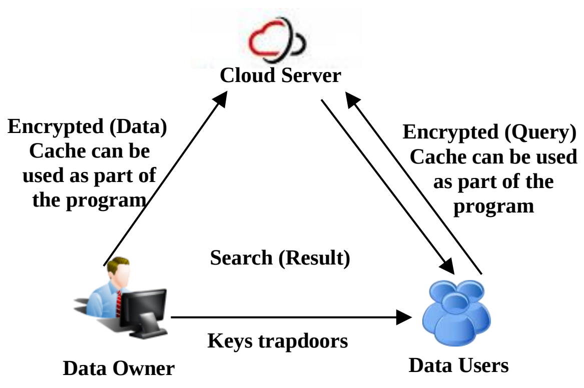

Cloud Server
Encrypted (Data) Cache can be used as part of the program
Encrypted (Query) Cache can be used as part of the program
Search (Result)
Keys trapdoors
Data Owner
Data Users

Fig. 1 System Architecture for Outsourced Cloud Data Services [7].

In a typical data mining scenario for outsourced cloud data services (as illustrated in Fig. 1), we can identify three different components [5]:

- Data Owner: this is the data provider, who owns some of the data that are required in the data mining task.
- Cloud server: this actor, known as the data miner, performs data mining tasks on the data.
- Data Users: is any employee, contractor or third-party provider who is authorized by the Data Owner to access information assets.
- Cache which can be part of the data processing program can be used between data user and cloud server or/and between the data owner and cloud server to reduce the computation time of the programs

Unless the system uses a cache partitioning mechanism, caches are shared among the tasks. As a consequence, the gains from using the caches can be reduced when some tasks run in parallel or preempt each other. It also means that the decisions made by a real-time scheduler will have an impact on the actual computation time of the jobs.
Caching has been used as a complementary architectural feature in our proposed Multi-broadcast Searchable Keywords Encryption (MBSKE) scheme, to improve the Quality-of-Service (QoS), reduce the overall network traffic, and to minimize the computational cost and complexity in a cloud environment [26].

This paper proposes the use of absorption time metric caching in a cloud server. This is defined as the time for which content remains cached in a node's cache. Absorption time depends on the cache replacement policy and the contents popularity distribution [6].

The data owner must achieve two tasks before sending their data to the cloud, i.e., encryption algorithms (confidentiality), and access control list (authorization [20]).

The searchable keyword encryption algorithm can be accomplished under the challenges of an authorized user (questioner) to explore those encrypted ones. This means that the authorized user (questioner) tries to investigate the encrypted data on the cloud without revealing the question, so the cloud server tries to search for all encrypted data.

The proposed MBSKE aims at reducing processing time and traffic by using cache algorithm. In this research, the movement of data between the cloud and users in terms of time and number of operations in the case of a specific file search is investigated. The results of our investigations demonstrate reductions in search time and collisions, thus indicating the suitability of the proposed scheme in big data environments.

Public key searchable encryption (PKSE) is an encryption scheme capable of providing privacy for all parties without revealing any information to the cloud [8].

Public-key broadcast encryption (BE) is used to broadcast encrypted data to all authorized users. Consider a set S using a unique public key. Any user in S is able to decrypt the message using their own private key, while users outside set S should not be able to do so even if they are part of the cloud.

Applying public key encryption with keyword encryption for a number of users helps in generating different cipher texts for various users, which leads to increasing the size of the database proportionally with the number of recipients, induces inflating of data issues, increases the traffic of packets over the network, generates more computational cost in the client sides and replicates encrypted data which could infect security [9].

Some applications require sending encrypted data to a number of users in a cloud environment. It has been indicated in [8] that it is possible to generate different ciphertexts for a number of legal users, which leads to increasing the size of database proportional with the number of users. In turn, this increases the traffic of packets over the network. Moreover, it leads to increasing computational cost in the client sides and replicates encrypted data infecting security, that is:

- Size of the encrypted data will be increased with the number of users with O(2n).
- Data owner needs to send new PKSE for every new user, which causes an increase in traffic between the data owner and cloud servers.
- Encryption of the same data on numerous occasions abuses security.

TABLE 1

Definition Types of Queries

Query Type	Source	Ciphertext Size
Equality query: $(xi = a)$ for any $a \in T$	[10, 8,22,23,32]	O(1)
Comparison query: $(xi \geq a)$ for any $a \in T$	[13,14]	$O(\sqrt{n})$
Subset query: $(xi \in A)$ for any $A \subseteq T$	[11,31]	O(n)
Comparison conjunction: $(x1 \geq a1) \wedge . . . \wedge (xw \geq aw)$	[11,30]	O(nw)
Subset conjunction: $(x1 \in A1) \wedge . . . \wedge (xw \in Aw)$	[11,30]	O(nw)
Disjunctive Keywords Search	[33]	$O(n^2)$

In a data mining environment, there are several types of queries (refer to Table 1). To the best of our knowledge, only simple equality queries on encrypted data are possible. In a nutshell, the proposed model aims to improve the traditional approach of searching the data in the cloud using a query (trapdoor), which contains a set of (encrypted) keywords.

The contributions of this research are as follows:

- Propose a suitable solution to the problem of searching encrypted query contains a set of keywords in encrypted data which lead to increasing the efficiency of the query processing.
- Provide searchable keyword encryption for multi-user environments with fixed size ciphertext instead of PEKS, whereas the size of ciphertext is O(2n) reduce the size of the server storage from O(2n×m) (if we use PKSE) or O(3×m) in BSKE to O(3+m), where n is number of recipients, and m is number of keywords.
- Reduce the traffic load of communication between the data owner and the cloud server.
- Reduce the complexity to improve the performance of searching.

The remainder of the paper organized as follows. In section II, we introduce some related works to the task scheduling problem that have been studied. In section III, our proposed architecture model will be described, as well as details of the problem formulation and presents our proposed method in which Sections 3.2 and 3.3, the first point of our contributions are discussed. Experimental results are shown in Section IV where the second and fourth points of our contributions are shown in Section 4.1, followed by our conclusions and suggestions for future work in section IIV.

2. Related Work

Data security is a serious concern when migrating data to a cloud Database Management Systems (CDBMS). Database encryption, where sensitive columns are encrypted before they are stored in the cloud, has been proposed as a mechanism to address such data security concerns. The intuitive expectation is that an adversary cannot "learn" anything about the encrypted columns, since she does not have access to the encryption key. However, query processing becomes a challenge either on cloud computing or NGI, since it needs to "look inside" the data [27]. Encrypted data querying has been the subject of numerous investigations. Song et al. [10] developed a mechanism for equality tests on data encrypted with a symmetric key system, whilst Boneh et al. [11] and Wu et al. [29] constructed equality tests in the public-key setting.

In most studies on database privacy, cloud computing and NGI focuses on two kinds of reduced keywords encryption methods. The first technique aims to reduce the size of the encrypted keyword in the database and the computational costs from the client side. The second method is to encrypt one keyword for many users in a group to avoid duplication of data as in previous schemes.

In [35], Guo, et al. focus on a new location privacy problem from a historical query point of view in Location-Based Services (LBS), however the paper did not consider the grid size which affects the attack.

Maheshwari, et al., [36], studied a framework to measure end-to-end traffic between Server and Client located in India and USA. But their proposed model should meet other networks demands, not only the networks between USA and India. In [37], the security analysis shows that the confidentiality of data and security of convergent keys, as well as protecting the user privacy. However, this study needs to provide consideration to the identity privacy of data owners. Yao, Xin, et al [38], explore the problem of privacy-preserving query for multi-source in the cloud-based PHR environment for Cloud-based Personal Health Record systems. While [39], presented a secure multi-attribute conjunctive keyword search scheme over encrypted cloud data. Both [38] and [39] did not consider the cost of data searching cost in the cloud, which is considered important aspects in our research methodology.

The researchers in [14] proposed novel concept of Key-Aggregate Searchable Encryption (KASE). This scheme supports searchable group data sharing functionality, which allows any user to share a group of specified files with a group of selected users, while allowing users to perform keyword search on the group of files. The researchers in [14] assumes that the cloud server is considered semi-honest, i.e. all servers would not conspire with other servers, but they are curious about the users' privacy and would mine information as much as possible and can only have access to some of query information, which is a limitation aspect in [14]. In [15], an efficient and shareable ID-based encryption with a keyword search (IDEKS) in the cloud is proposed. Here, the server holds the master key. The master key has the ability to generate trapdoors as a receiver in IDEKS by forging queries, but the limitation of this scheme is that the number of master keys will be increased as the number of the users and queries will be increased.

In [16], a novel framework, which allows authorized users to perform keyword-based searches on encrypted data, is proposed. This does not require sharing a unique secret key. One of the features of this method is that it provides two-layered access control, thus denying access to the shared data in the case of unauthorized users.

In [19], a technique for authorized searchable encryption with attribute-hiding was proposed, which enables authorized users to perform a keyword search and then decrypt ciphertext. This is the first contribution to integrate (PEKS)[11], (PEKSL) [29], with the Ciphertext-Policy Attribute-Based Encryption (CP-ABE) based lattices assumption. In contrast to previous solutions [17-18], their scheme achieves attribute-hiding, which could prevent the access to sensitive user information. The security of the scheme is based on the well-known Learning with Errors Assumption (LWE) [28]; meanwhile, data owners can sort the ciphertext. If the users want to extract the ciphertext from a time point, they only need to submit the trapdoor corresponding to the keyword in the cloud server as illustrated in Fig.2.

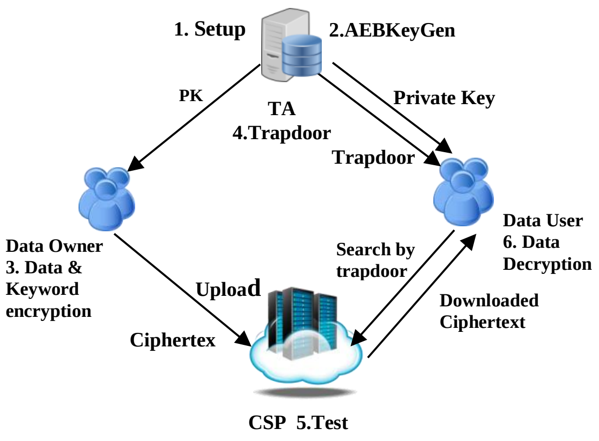


Fig. 2 System architecture of (ASE) Adaptive Server Enterprise in cloud computing [17].

Ali. et al., [20], studied searchable keyword encryption in the multiuser scenario with fixed size ciphertext of O(3n). They semantically proved that the security against adaptive keyword can adverse random oracle model under Bilinear Diffie-Hellman Summation Exponent (n-BDHSE) assumption. Their model only considered computational costs on the client's side and ignored cloud side computations. Moreover, their model was only designed for single keyword queries. This work provided the motivation for our proposed method to improve traffic between the clients and the cloud.

3. Proposed Model

With the increased number of broadcast users and the rapid growth of the amount of information, sending quires and receiving encrypted data from the cloud are becoming slow processes using a set of keywords. This set of keywords is sent from the users to the cloud server in an encrypted form, thus hiding all information about the user or the content of the queries from the cloud server. Table 2 shows the notations and definitions for our proposed pseudocode MBSKE.

TABLE 2
The notation and their definitions for our proposed Multi-broadcast Searchable Keywords Encryption scheme

W	The plaintext document collection, denoted as a set of m documents W=(W_1,W_2,W_3, … W_M).
E	The encrypted document collection stored in the cloud server, E=(E_1,E_2,E_3,…..,E_m).
W	The keywords in the query w=(w_1,w_2,w_3….,w_m).
T	The trapdoor for the search request wm
E_w	The identity collection of documents returned to the search user Ew=(E_{w1},E_{w2},E_{w3}….,E_{wm})
λ	Security parameter
P_k	a Public key
d_k	a Private key
u_i	Legal users
S	Group of the users

3.1. Bilinear mapping

Bilinear pairing maps are the basis of the MBSKE scheme [21]. This uses an elliptic curve to map two groups $e: G \times G \rightarrow G_T$ in the case of symmetric groups $e: G \times G \rightarrow G_T$.

Both groups have the same prime order group. The following properties must be satisfied:

1. *Bi-linearity*: For all $u,v \in G$ and $a,b \in Z^*_p$
   we have $e(u^a, v^b) = e(u,v)^{ab}$
2. *Non-degeneracy*: $e(g,g) \neq 1$
3. *Computability*: There is an efficient algorithm to compute *e(u,v) for any u,v* ∈ g.

3.2. Security

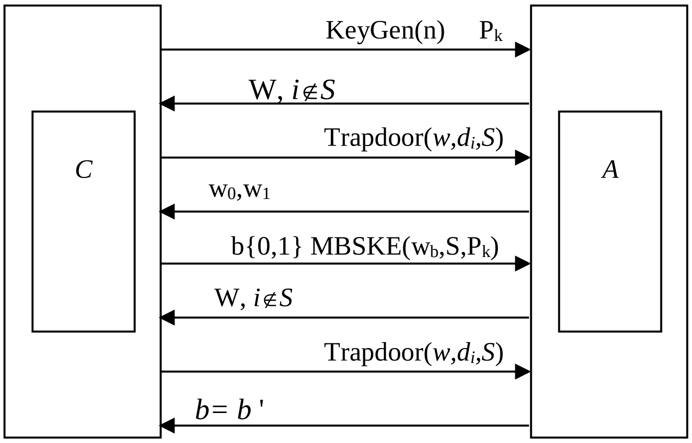


Fig. 3 Security model

Fig.3 shows the steps of security model. In this case, in step1, *A (the attacker)* selects an unauthorized user $i \notin S$ in which *A* aims to attack, where *A* chooses any keyword *w* and asks *C (the challenger)* to provide *Ti(w). C* will then respond by finding the private key for the unauthorized user $i \notin S$, computes *Ti(w),* and send it to *A*, *A* will continue asking *C* for other keywords .
When *A* finished all queries of trapdoors, it will then intend to send two equal length keywords *w0,w1* to *C*, then tosses fair flop coin and chooses w_b and computes MBSKE*s* (w_b) by running MBSKE algorithm for *S (*authorized set of users) and sends it to *A* as challenge .
Step2, the attacker *A* does the same thing continuously for polynomial times as in query step 1, but with the restriction that keywords w_0,w_1 cannot be queried any more. So the attacker A returns a guess b' to MBSKE. If b= b', it means *A* wins the game, the algorithm outputs 1. Otherwise fails and outputs 0.

3.3. Multi Broadcast Searchable Keywords Encryption (MBSKE) Scheme Model

The proposed Multi Broadcast Searchable Keywords Encryption (MBSKE) model as illustrated in Fig.3 requires the following assumptions:
Consider the data owner (Alice) wants to encrypt the same data under a master public key for a group of users and stores this encrypted data with a group of authorized users. Malice is one of those recipients who will use his private key to ask Alice whether or not Alice has stored encrypted data. Alice will then search in all encrypted data using the master public key. If Alice finds the matching keywords, the answer will be “yes”, otherwise the answer will be “no”. It is expected that Alice will learn nothing from the encrypted data and the query.
Broadcast searchable keywords encryption (BSKE) [20] is used as the basis for our proposed MBSKE with queries contains a number of fields:

- Use a caching algorithm to divide the data (multi-field queries) over encrypted data.
- Reduce the complexity
- Reduce the implementation time in the trapdoor and searching.
- Reduce the traffic between the cloud server and clients.

- Propose a suitable solution to the problem of searching encrypted query contains many fields in encrypted data.

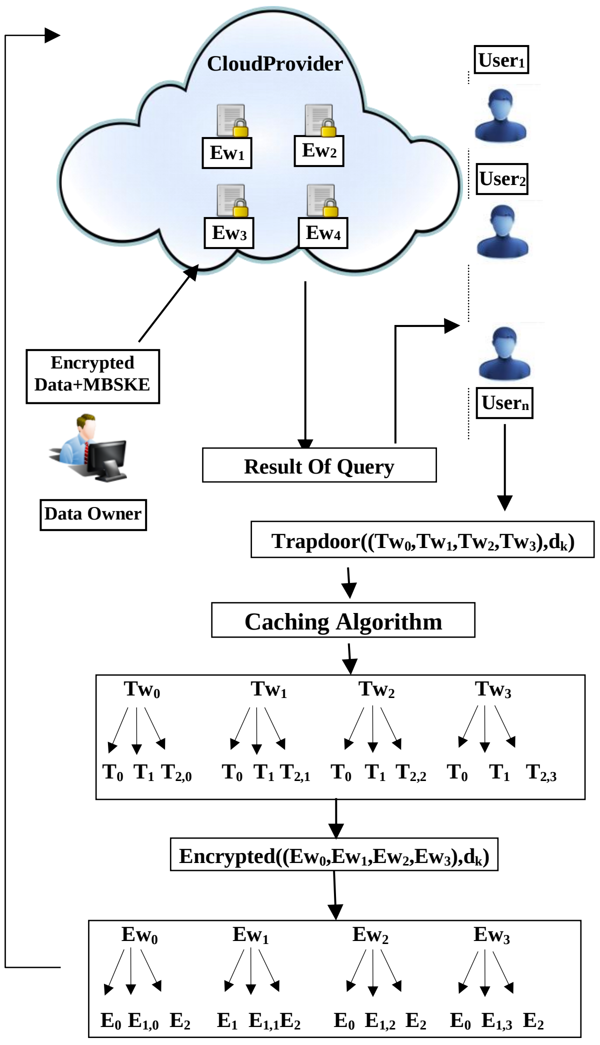

CloudProvider
Ew1
Ew2
Ew3
Ew4
User1
User2
Usern
Encrypted Data+MBSKE
Data Owner
Result Of Query
Trapdoor((Tw0,Tw1,Tw2,Tw3),dk)
Caching Algorithm
Tw0 Tw1 Tw2 Tw3
T0 T1 T2,0 T0 T1 T2,1 T0 T1 T2,2 T0 T1 T2,3
Encrypted((Ew0,Ew1,Ew2,Ew3),dk)
Ew0 Ew1 Ew2 Ew3
E0 E1,0 E2 E1 E1,1E2 E0 E1,2 E2 E0 E1,3 E2

Fig. 4 The proposed MBSKE System Model

3.4. Detailed Design of MBSKE Scheme

The proposed scheme as illustrated in Algorithm 1 consists of the following functions:

- ***Setup*** (λ, n, m): The setup algorithm takes in the security parameter λ as input, m is the number of keywords in query, n is the number of users, and provides the output P_k which is the public key.
- ***KeyGen*** (S, P_k, d_k): Given a master key "P_k" and a unique identity "d_k" representing a user u_i in the group of S users, this function computes the public/secret key for user ui, which is sent through a secure channel.
- ***Cache*** (Tw_m, k, S, P_k): This algorithm is run by the cloud server to adjust the aggregate trapdoor "Tw_m" and generates a set of separate trapdoors for each keyword in the query. The Cache algorithm is used to reduce the repetition of some of the data, coming from the users to the cloud (trapdoor). In this algorithm, the cloud will divide the trapdoor to m number of separate small trapdoors according to the number of keywords in the query,

in which each trapdoor = $Tw_m(T_0,T_1,T_{2,q})$. The cache algorithm is used to fix the parameters T_0,T_1 for m trapdoors. This will result in savings with respect to the computational times for the search.

- ***MBSKE*** (w_m, P_k, S) = MBSKE (E_0 , $E_{1,q}$, E_2): This algorithm computes the encrypted keyword w_m. The data owner inputs the group set, master public key P_k and the keyword w_m algorithm outputs MBSKE which consists of three parts : $w_m=\{E_0,E_{1,q}, E_2\}$=MBSKE(P_k,S) for each attribute.

- ***Trapdoor*** (d_k , k , S , w_m) = Twm: This algorithm is executed by the group of authorized users that inputs the keywords w_m, the private key d_k, the k index of the user and the group S, and then it computes the trapdoors, Twm=$\{T0,T_1,T_{2,q}\}$=Trapdoor(d_k,k,S,w_m) The output of this function will be sent to the server as a query request.
  In trapdoor, authorized users will query the cloud server whether it stored the encrypted keyword using their own private key and in turn, this will compute the trapdoor of the keyword as shown in Equation 2. Then the user will sends the tuple of the trapdoor to the cloud server.
- ***Decision*** (Pk, S, MBSKE,TWm): This algorithm is used by the cloud server to examine the validity of the MBSKE and trapdoor query. The cloud server will search and match all MBSKEs in the encrypted data on trapdoors. Then it will respond by “yes” or “no”, accordingly.

Algorithm 1: Main proposed MBSKE scheme

```
Setup ( $\lambda$,n,m)
Begin
   n = #S, S is the group of legitimate users
   m = Number of keywords in the query
   Trapdoor($d_k$,k,S,$w_m$)
   For each $w_m$ create counter count q
        Where q ϵ [m] and m ⊆ M
         M is the number of all encrypted files in the cloud
         For initial setup count = 0
            Search for matching keyword,
            If there is matching, then
               {
                  count = count + 1
                  MBSKE = MBSKEm
                  Decision(Pk,S,MBSKE,Twm) = yes
               }
            Else jump to the next record
               Continue until the last record
            Continue until the last attribute
End.
```

It is considered that the data owner sends the encrypted data with MBSKE to the cloud server. The Database X contains a number of encrypted files $E_{Wq}=(E_{W1}, E_{W2}, E_{W3}, E_{W4} ,…. , E_{WM}, S, p_k)$, and a set of authorized users which send a query to the cloud server $Tw_m= (Tw_1,Tw_2,Tw_3,…….,Tw_m,d_k)$.

Suppose we have n users and m encrypted files then:

1. The data owner encrypts files ($W_1,W_2,…,W_M$) under the authorized set S and then computes MBSKE (W_M,P_k,S) = $\{E_0,E_{1,q},E_2\}$, $\{E_q \in G_T^{3\times m}, i = 1,2,.., m\}$, G_T is the symmetric group mapping of two groups, and outsourcing the encrypted data in third party (CS).
2. The users $u_x \in S$ (Decision maker DM) generally refer to those who are authorized to search for encrypted keywords in the encrypted data, he/she uses his/her private key pr_{u_x} to compute the trapdoors. The trapdoors consist of many

encrypted words (w_1, w_2, ….,w_m). The private key is indexed by $q \in [wm]$ and obviously, Where a number of queries features are $w_m \leq M$,let index is $\{q_1, q_2, \dots, q_m\}$, M is the number of all encrypted files in the cloud.

3. After sending the trapdoor to the cloud, the caching algorithm divides the trapdoor Tw to many separate trapdoors (Tw_1,Tw_2,…..Tw_m), while keeping some fixed parameter values to avoid repetition of the data, and reduce the computational cost. The procedure is as follows:

   Trapdoor(d_k,k,S,w_m)=Tw_m(T_0,T_1,$T_{2,q}$)
   where $Tw_m \in G_T^{3\times wm}$
   T_0 refers to the private key (fixed value), T_1 refers to the public key (fixed value), $T_{2,q}$ refers to the encrypted keyword.
4. Testing the query q with features f_n as
   Decision (P_k,S,MBSKE,Tw_1)
   Decision (P_k,S,MSKE,Tw_2)
   …
   …
   Decision (P_k,S,MBSKE,Tw_m)

The sender (data owner) inputs the keywords W, master public key and authorizes set S then chooses a random t ϵ Z_p and computes

$$\text{MBSKE (wm,Pk, S)} = \{E_0 , E_{1,q} , E_2\} \qquad (1)$$

$$E_0 = g^t$$

$$E_{1,q} = \left(H(w_q)\right)^t$$

$$E_2 = (v.\prod_{j\in S} g^{a^{n+1+j}})^t$$

Then sends this data to the cloud server.

For trapdoor algorithm, the legitimate clients ask the cloud server whether it stored the encrypted keywords by using their own private keys and computes trapdoor of the keyword firstly the client chooses a random r ϵ Z_p and computes

$$Trapdoor(d_k, k, S, w_m) = \{T_0, T_1, T_{2,q}\} \qquad (2)$$

$$T_0 = g^r$$

$$T_1 = \left(g^{(a^k)}\right)^r$$

$$T_{2,q} = \left(d_k.\left(H(w_q)\right).\prod_{j\in S} g^{a^{n+1-j+k}}\right)^r$$

This sends a tuple to the cloud for validation of the encrypted keywords W.

$$Decision\ (Pk,S,MBSKE,Twm) = \frac{e(T_{2,q},E_0)}{e(T_1,E_2)} = e\left(T_0, E_{1,q}\right) \quad \frac{e(T_{2,q},E_0)}{e(T_1,E_2)} = \frac{e\left(\left(d_k.\left(H(w_q)\right).\prod_{j\in S} g^{a^{n+1-j+k}}\right)^r, g^t\right)}{e\left(\left(g^{(a^k)}\right)^r, (v.\prod_{j\in S} g^{a^{n+1+j}})^t\right)}$$

$$= \frac{e\left(\left(v^{(a^k)}.\left(H\left(w_q\right)\right).\prod_{j\in S} g^{a^{n+1-j+k}}\right)^r, g^t\right)}{e\left(\left(g^{(a^k)}\right)^r, (v.\prod_{j\in S} g^{a^{n+1+j}})^t\right)}$$

$$= \frac{e\left(\left(v.\prod_{j\in S} g^{a^{n+1-j}}\right)^{(a^k)^r}, g^t\right).e\left(\left(H\left(w_q\right)\right)^r, g^t\right)}{e\left(\left(g^{(a^k)}\right)^r, (v.\prod_{j\in S} g^{a^{n+1+j}})^t\right)}$$

$$= e\left(\left(H(w_q)\right)^t, g^r\right) = e\left(E_{1,q}, T_0\right)$$

4. Simulation Result and Discussion

4.1. Simulator setup

We conducted extensive simulation experiments to evaluate the proposed MBSKE. The evaluation was carried out on a Dell Inspiron E450 with 240 GHz Intel Processor core 2 Due, OS is 64-bit Windows 7.
For comparison, we implemented the MBSKE, BSKE and PEKSL schemes, as well as the encryption and decryption for hash functions used AES 128 bit in C++ .net with MIRACL library. We also used MATLAB for data traffic testing.

4.2. *Processing time*

For trapdoor and runtime, we used 10 sets and 2 different size of queries (10 sets of users) * 2 (type of queries) * 2 (metrics trapdoor and runtime) * 3 (No. of schemes) = 120 experiments for tables 3 and 4.

TABLE 3
Simulation Assumptions and Parameters

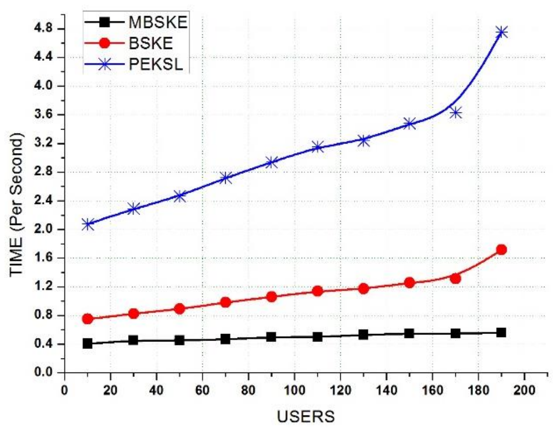
Parameter	Value
Keywords	1,3,5
#Users	20 – 200
Group of authorized users	8
Type of encryption	AES128 (type of the Encryption)
Reference scheme	BSKE, PEKSL
Proposed scheme	MBSKE

Because of our scheme in broadcast environment it is important to consider different numbers of users to measure the performance of our scheme. In the paper we assume the size of sets of user start from 20 users to 200. We can assume another numbers of users.
In addition, we considered the number of users as assumption because we are working in broadcast, so we considered different numbers of users. The most effect here in the number of keywords in the query.

TABLE 4
Trapdoor Time and Runtime

No. of Users	Trapdoor (3 Keywords)			Runtime (3 Keywords)		
	MBSKE	BSKE	PEKSL	MBSKE	BSKE	PEKSL
10	0.39	0.546	1.564	0.343	0.577	1.309
30	0.406	0.608	1.742	0.374	0.624	1.416
50	0.421	0.671	1.922	0.375	0.687	1.559
70	0.452	0.718	2.057	0.406	0.733	1.663
90	0.453	0.764	2.189	0.405	0.858	1.947
110	0.484	0.826	2.366	0.436	0.811	1.84
130	0.484	0.874	2.504	0.452	0.921	2.089
150	0.515	0.936	2.682	0.452	0.98	2.223
170	0.561	1.014	2.905	0.484	0.967	2.194
190	0.546	0.999	2.862	0.515	1.139	2.584

No. of Users	Trapdoor (5 Keywords)			Runtime (5 Keywords)		
	MBSKE	BSKE	PEKSL	MBSKE	BSKE	PEKSL
10	0.405	0.749	2.0747	0.406	0.78	2.1824
30	0.453	0.827	2.2908	0.405	0.842	2.3559
50	0.452	0.89	2.4653	0.437	0.909	2.5434
70	0.468	0.983	2.7229	0.487	1.045	2.9239
90	0.499	1.061	2.939	0.515	1.03	2.8819
110	0.5	1.139	3.1550	0.52	1.123	3.1422
130	0.53	1.17	3.2409	0.5344	1.202	3.3632
150	0.548	1.257	3.4819	0.554	1.273	3.5619
170	0.546	1.311	3.6315	0.561	1.339	3.7465
190	0.562	1.716	4.7533	0.568	1.716	4.8014



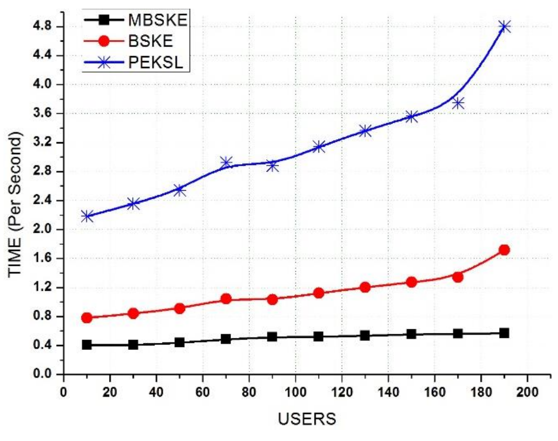


Fig 5.a Trapdoor time searching (5 keywords) **Fig 5.b Run time searching (5 keywords)**

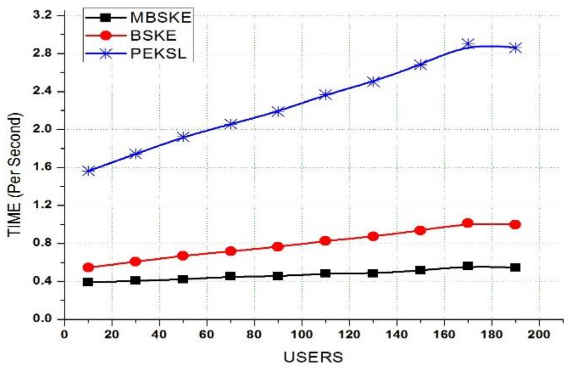


Fig 6.a Trapdoor time searching (3 keywords)

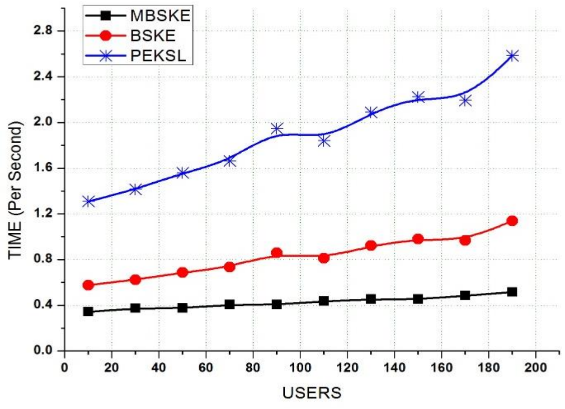


Fig 6.b Run time searching (3 keywords)

Figures 5 and 6 show the effect of using the cache algorithm in MBSKE Eq(1), whereas the schemes BSKE and PEKSL refer to the models without using the cache algorithm, which uses all the parameters, elements, and the mechanism of sending the encrypted queries from the users to the cloud and receiving the response of the queries from the cloud to the users, in encryption mode as well. Furthermore, the MBSKE scheme shows the same model using the cache algorithm.

The performances of the schemes are benchmarked against varying number of users and keywords as shown in Table 3. In terms of efficiency, Fig. 5 shows that the trapdoor Eq(2), and execution times of the proposed MBSKE model is less than the schemes BSKE and PEKSL. The proposed MBSKE model also outperforms in the case of 3 keyword size as shown in Fig 6. The parameters (T_0, T_1) of the proposed model did not repeat the calculations more than once hence its running time is less than BSKE and PEKSL schemes.

According to Tables 3 and 4, and the result as shown in Figure 4, the trapdoor times in MBSKE are better than in BSKE and PEKS schemes, as it takes less time to implement the trapdoor time.

The following equation calculates the percentage of improvement for trapdoor time and time of run search as the following:

$$Percentage\ of\ Improvements = \sum_{i=1}^{n} \left(\frac{|X1_i - X2_i|}{X2_i} \right) \times 100 \qquad (3)$$

Where $X1_i$ refers to the MBSKE trapdoor time and time of run search, $X2_i$ refers to the BSKE trapdoor time and time of run search and n is the number of users.

To evaluate the proposed method, it is simulated using Matlab and compared with two existing schemes, a BSKE and PEKS. The results show that the proposed scheme MBSKE has 40% improved trapdoor time and search time in comparison with BSKE and improved performance in comparison to PEKSL.

In addition, the performance of searching in the MBSKE scheme is higher than the BSKE and PEKSL schemes (refer to Table 4) due to the decreased repetitions in the calculations of certain parameters.

The processing time of the proposed scheme and the benchmarked BSKE and PEKSL is given as:

- MBSKE and BSKE complexity = O(3) for one word searching since there is no caching in this case, 3 refer to the private key, public key and the encrypted word .
- For multi keywords, BSKE complexity = O(3m), MBSKE complexity = O(2+m), whereas PEKSL complexity = O(3mn).

For instance, if m = 5 then:

BSKE complexity = 15, MBSKE complexity = 7

The general idea of the schemes MBSKE, BSKE and PEKSL is that, how to send the encrypted queries from the users to the cloud and receiving the response of the queries from the cloud to the users, in encryption mode as well. The main used parameters on these scheme are the run time, the time of processing the queries the cost of communication and throughput. All of these schemes used different technique to improve the performance. In addition, MBSKE, BSKE are working in broadcast environment. Therefore the performance is better than PEKSL with increasing the users. But the scheme MBSKE appear more efficient when the query contains more than one keyword, because of MBSKE the cache algorithm during sending the encrypted query from the user to the cloud provider.

4.3. Throughput

Throughput is a very important parameter that helps measure the traffic between the server and clients to decide on the number of servers/hardware required for SharePoint implementation. Throughput is the number of operations/requests that a server can handle per second. It is measured in requests per second (RPS). RPS measurements can be converted to the total number of users by using a model based on typical user behaviour.

- *Concurrency*: the percentage of users that are actively using the system.
- *Request rate*: the average number of requests per hour

There are many tools to measure and analyse the traffic which can give report about the traffic such as traffic type, machines and so on. In the paper we considered some of parameters to measure and analyse the traffic according to the assumptions in Table 5 in which:

users = [500, 1000,1500,2000,2500,3000,3500,4000];
Th = Throughput (# of requests per second)
PAU = Percentage of active users
RR = Request Rate
Keywords = [5-10-15-20-25-30]
In the proposed model, throughput is calculated as:
# Active Users = 10% of # Users
Request Rate (RR) = 30 * 3600
Time per keyword search = 1 Second
TU= total users
Whereas RST refer to (Request service time) which is the Time per Keyword search.

$$TL= \sum_{i=1}^{m} \left(\frac{\#TU \ \times \ AU \times keywords}{RR_i \ \times \ (RST_i)} \right) \qquad (4)$$

TABLE 5

Simulation Assumption and Parameters for Traffic

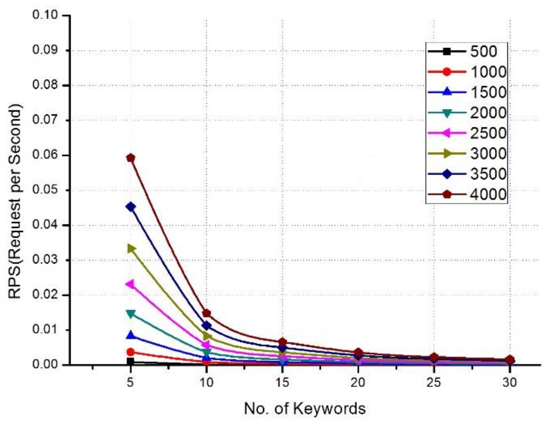
Parameter	Value
Keywords	[5-10-15-20-25-30]
# users	[500,1000,1500,2000,2500,3000, 3500,4000]
# Active Users	10% of # Users
Request Rate (RR)	30 * 3600
Time per keyword search	1 Second

In order to evaluate the performance of the proposed MBSKE algorithm in reducing the traffics and increasing resources utilization, we have utilised various scenarios. In the first scenario, we used 5 keywords with different numbers of users [500-4000], then we used 10 keywords with the same numbers of users, and so on. Table 6 shows the throughput comparison results among BSKE and MBSKE algorithms.

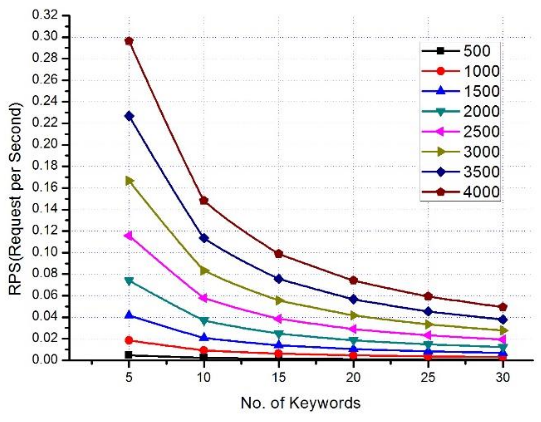
TABLE 6

Throughput of BSKE and MBSKE

NO. of Keywords	NO. of Users															
	500		1000		1500		2000		2500		3000		3500		4000	
	BSKE	MBSKE	BSKE	MBSKE	BSKE	MBSKE	BSKE	MBSKE	BSKE	MBSKE	BSKE	MBSKE	BSKE	MBSKE	BSKE	MBSKE
5	0.00093	0.00463	0.0037	0.01852	0.00833	0.04167	0.01481	0.07407	0.02315	0.11574	0.03333	0.16667	0.04537	0.22685	0.05926	0.2963
10	0.00023	0.00231	0.00093	0.00926	0.00208	0.02083	0.0037	0.03704	0.00579	0.05787	0.00833	0.08333	0.01134	0.11343	0.01481	0.14815
15	0.00010	0.00154	0.00041	0.00617	0.00093	0.01389	0.00165	0.02469	0.00257	0.03858	0.0037	0.05556	0.00504	0.07562	0.00658	0.09877
20	0.00006	0.00116	0.00023	0.00463	0.00052	0.01042	0.00093	0.01852	0.00145	0.02894	0.00208	0.04167	0.00284	0.05671	0.0037	0.07407
25	0.00004	0.00093	0.00015	0.0037	0.00033	0.00833	0.00059	0.01481	0.00093	0.02315	0.00133	0.03333	0.00181	0.04537	0.00237	0.05926
30	0.00003	0.00077	0.00010	0.00309	0.00023	0.00694	0.00041	0.01235	0.00064	0.01929	0.00093	0.02778	0.00126	0.03781	0.00165	0.04938



Fig(7.a) Throughput (BSKE)



Fig(7.b) Throughput (MBSKE)

According to assumptions in Table 5 and the results in Table 6, Fig 7 shows the performance level of the cloud. The throughput rate of MBSKE model provides more efficiency Eq. (4), even with increasing the number of users as compared to BSKE model.

4.4. Communication overhead and the cache algorithm

The communication cost of user's trapdoor is a challenging aspect in cloud computing in which we used caching algorithm to enhence the time as illustrated in Figure 4.
We used the cache algorithm to reduce the repetition of encrypted data; in trapdoor to split the query to *m* trapdoors according to the number of keywords in the query, whereas each trapdoor $Tw_m(T_0,T_1,T_{2,q})$. The cache algorithm is used to fix the parameters T_0,T_1 for *m* trapdoors to avoid the repetition of data. So the computational time of searching will be less.

5. CONCLUSIONS

Data security is a serious concern when migrating data to a cloud Database Management Systems. Database encryption, where sensitive columns are encrypted before they are stored in the cloud, has been proposed as a mechanism to address such data security problem.
In this paper, we proposed Model Based on Multi Broadcast Searchable Keywords Encryption to process queries with a set of keywords. This set of keywords is sent from the users to the cloud server in an encrypted form, thus hiding all information about the user or the content of the queries from the cloud server.
On the basis of all carried experiments, we concluded that our caching algorithm in MBSKE scheme has a significant advantage over the schemes on the overall performance such as reducing the query processing and traffic load.

By adopting this scheme, cloud servers can be optimally utilized with the possibility to reduce the number of the cloud resources for the same task. Furthermore, it is noted that the use of cache has contributed positively in the process of sending the query. In addition, we provided less computational cost on the clients side and enhanced scheme in the cloud side.

ACKNOWLEDGMENT

This work was supported by NSFC key project under grant No.61731017 and the 111 Project under Grant No.111-2-14.

REFERENCES

[1]. Xu, L., Weng, C. Y., Yuan, L. P., Wu, M. E., Tso, R., & Sun, H. M. "A shareable keyword search over encrypted data in cloud computing." The Journal of Supercomputing, 1-23. (2015).
[2]. Zou, X., Xiang, J.: Dynamic broadcast encryption scheme with revoking user. Wuhan Univ. J. Nat. Sci. 18(6), 499–503 (2013)
[3]. Security management in health using ISO27799:2016
[4]. Popovich, Vasily V., et al. "Information Fusion and Geographic Information Systems." Lecture Notes in Geoinformation and Cartography. Proceedings of the Fourth International Workshop. 2009.
[5]. Xu, L., Jiang, C., Wang, J., Yuan, J., &Ren, Y. "Information security in big data: privacy and data mining." Access, IEEE, 2, 1149-1176.(2014).
[6]. Ioannou, Andriana, and Stefan Weber. "A survey of caching policies and forwarding mechanisms in information-centric networking." IEEE Communications Surveys & Tutorials 18.4 (2016): 2847-2886.
[7]. Goldreich, O., &Ostrovsky, R. "Software protection and simulation on oblivious RAMs". Journal of the ACM (JACM), 43(3), 431-473.(1996).
[8]. Boneh, D., Di Crescenzo, G., Ostrovsky, R., &Persiano, G. "Public key encryption with keyword search". In Advances in Cryptology-Eurocrypt 2004 (pp. 506-522). Springer Berlin Heidelberg.(2004, May).
[9]. P. Mell and T. Grance, "The NIST definition of cloud computing" National Institute of Standards and Technology, vol.53,no. 6, 2009.
[10]. Song, D. X., Wagner, D., &Perrig, A. "Practical techniques for searches on encrypted data". In Security and Privacy, 2000. S&P 2000. Proceedings. 2000 IEEE Symposium on (pp. 44-55). IEEE. (2000).
[11]. Boneh, D., & Waters, B. "Conjunctive, subset, and range queries on encrypted data". In Theory of cryptography (pp. 535-554). Springer Berlin Heidelberg.(2007).
[12]. Dan Boneh, AmitSahai, and Brent Waters. "Fully collusion resistant traitor tracing with short ciphertexts and private keys. In Eurocrypt". '06, (2006).
[13]. Boneh, D., & Waters, B. "A fully collusion resistant broadcast, trace, and revoke system". In Proceedings of the 13th ACM conference on Computer and communications security (pp. 211-220). ACM.(2006, October).
[14]. Cui, B., Liu, Z.,& Wang, L. "Key-aggregate searchable encryption (KASE) for group data sharing via cloud storage". (2015).
[15]. Li, M., Yu, S., Lou, W., &Hou, Y. T. "Toward privacy-assured cloud data services with flexible search functionalities". In Distributed Computing Systems Workshops (ICDCSW), 2012 32nd International Conference on (pp. 466-470). IEEE.(2012, June).

[16]. Li, J., Li, J., Liu, Z., &Jia, C. “Enabling efficient and secure data sharing in cloud computing. Concurrency and computation: practice and experience”, 26(5), 1052-1066.(2014).
[17]. Lai, J., Zhou, X., Deng, R. H., Li, Y., & Chen, K. “Expressive search on encrypted data”. In Proceedings of the 8th ACM SIGSAC Symposium on Information, computer and communications security (pp. 243-252). ACM.(2013).
[18]. Lv, Z., Hong, C., Zhang, M., &Feng, D. “Expressive and secure searchable encryption in the public key setting”. In Information Security (pp. 364-376). Springer International Publishing.(2014).
[19]. Zeng, F., &Xu, C. “A Novel Model for Lattice-Based Authorized Searchable Encryption with Special Keyword”. Mathematical Problems in Engineering.(2015).
[20]. Ali, M., Ali, H., Zhong, T., Li, F., Qin, Z., & Ahmed Abdelrahaman, A. A. “Broadcast Searchable Keyword Encryption”. InComputational Science and Engineering (CSE), 2014 IEEE 17th International Conference on (pp. 1010-1016). IEEE. (2014, December).
[21]. Lewko, A., Okamoto, T., Sahai, A., Takashima, K., & Waters, B. “Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption”. In Advances in Cryptology–EUROCRYPT 2010 (pp. 62-91). Springer Berlin Heidelberg.(2010).
[22]. Abdalla, Michel, et al. "Searchable encryption revisited: Consistency properties, relation to anonymous IBE, and extensions." Journal of Cryptology 21.3 (2008): 350-391.
[23]. Ostrovsky, Rafail. Software protection and simulation on oblivious RAMs. Diss. Massachusetts Institute of Technology, Dept. of Electrical Engineering and Computer Science, 1992.
[24]. Wang, Peishun, et al. "Privacy-preserving protocol for service aggregation in cloud computing." Software: Practice and Experience 42.4 (2012): 467-483.
[25]. Garg, Saurabh Kumar, and Rajkumar Buyya. "An environment for modeling and simulation of message-passing parallel applications for cloud computing." Software: Practice and Experience 43.11 (2013): 1359-1375.
[26]. Chéramy, Maxime, A. M. Déplanche, and P. E. Hladik. Simulation of Real-Time Multiprocessor Scheduling Using DES. Simulation and Modeling Methodologies, Technologies and Applications. Springer International Publishing, 2015:37-53.
[27]. Arasu, A, et al. "Querying encrypted data." IEEE, International Conference on Data Engineering IEEE, 2014:1259-1261.
[28]. Özgür Dagdelen, S. Gajek, and F. Göpfert. "Learning with Errors in the Exponent." International Conference on Information Security and Cryptology Springer International Publishing, 2015:69-84.
[29]. Daini Wu, Xiaoming Wang, and Qingqing Gan, “Public Key Encryption with Keyword Search from Lattices in Multiuser Environments,” Mathematical Problems in Engineering, vol. 2016, Article ID 6549570, 7 pages, 2016.
[30]. Zhang, Yu, Yin Li, and Yifan Wang. "Conjunctive and Disjunctive Keyword Search over Encrypted Mobile Cloud Data in Public Key System." Mobile Information Systems 2018 (2018).
[31]. Huang, Kaibin, Raylin Tso, and Yu-Chi Chen. "Somewhat semantic secure public key encryption with filtered-equality-test in the standard model and its extension to searchable encryption." Journal of Computer and System Sciences 89 (2017): 400-409.
[32]. Miao, Yinbin, et al. "VKSE-MO: Verifiable keyword search over encrypted data in multi-owner settings." Science China Information Sciences 60.12 (2017): 122105.
[33]. Zhang, Yu, Yin Li, and Yifan Wang. "Efficient Public Key Encryption with Disjunctive Keywords Search Using the New Keywords Conversion Method." Information 9.11 (2018): 272
[34]. Okamoto, T.; Takashima, K. “Achieving short ciphertexts or short secret-keys for adaptively secure general inner-product encryption”. Des. Codes Cryptogr. 2015, 77, 138–159.
[35]. Guo, Mingming, et al. "Query-Aware User Privacy Protection for LBS over Query-Feature-based Attacks." 2018 IEEE Symposium on Computers and Communications (ISCC). IEEE, 2018.
[36]. Maheshwari, Sumit, Sudipta Mahapatra, and C. S. Kumar. Measurement and Forecasting of Next Generation Wireless Internet Traffic. No. 525. EasyChair, 2018.
[37]. Liu, Ling, Yuqing Zhang, and Xuejun Li. "KeyD: Secure Key-Deduplication with Identity-Based Broadcast Encryption." IEEE Transactions on Cloud Computing (2018).
[38]. Yao, Xin, et al. "Privacy-preserving search over encrypted personal health record in multi-source cloud." IEEE Access 6 (2018): 3809-3823.
[39]. Zhang, Lili, Yuqing Zhang, and Hua Ma. "Privacy-preserving and dynamic multi-attribute conjunctive keyword search over encrypted cloud data." IEEE Access 6 (2018): 34214-34225.